FROM CHRONOLOGICAL NETWORKS TO BAYESIAN MODELS: CHRONOLOG AS A FRONT-END TO OXCAL

E Levy[1]* • E Piasetzky[1] • A Fantalkin[1] • I Finkelstein[2,1]

[1]Tel Aviv University, Archaeology and Ancient Near Eastern Cultures, Tel Aviv 69978, Israel
[2]School of Archaeology and Maritime Cultures, University of Haifa, Mount Carmel, Haifa 3498838, Israel

ABSTRACT. We present a new method for creating an OxCal Bayesian model that bypasses the complex task of writing OxCal code. Our methodology employs the recent CHRONOLOG software as a graphical front-end for generating OxCal scripts. This approach enables archaeologists to create complex Bayesian models—including *termini post* and *ante quem*, duration bounds and synchronisms—with the help of a user-friendly interface. The target audience can be divided into beginners, who might struggle to create chronological models using OxCal directly, and experienced OxCal users, who should find that CHRONOLOG saves time when coding complex models. Three case-studies from recent publications are presented.

KEYWORDS: Bayesian modeling, ChronoLog, chronology, OxCal, radiocarbon dating.

1. INTRODUCTION

This paper presents a method for creating OxCal Bayesian models without going through the complex process of writing OxCal code. Our approach uses the recently published CHRONOLOG software (chrono.ulb.be; Levy et al. 2021) as a graphical user interface (GUI) to generate OxCal scripts. It enables users with no training in programming or mathematics to easily build Bayesian models, even intricate ones, with the help of a user-friendly interface. The generated scripts can include *termini post* and *ante quem*, duration bounds, and several types of synchronisms. To the best of our knowledge, no GUI currently exists that automatically creates Bayesian OxCal models. In 2015, in a work comparable to ours, Dye and Buck showed how archaeological sequence diagrams, such as Harris matrices, can be converted to Bayesian chronological models. Their work was presented as a "proof of concept for the design of a front-end for Bayesian calibration software that is based directly on the archaeological stratigrapher's identification of contexts" (Dye and Buck 2015: 84). Such a "front-end for Bayesian calibration software" is the purpose of this paper, though our model is not specific to archaeological sequence diagrams, but rather uses generic chronological units, which can represent diverse realities, such as stratigraphical entities, historical reigns, and cultural phases. Graphical interfaces for producing Bayesian models without requiring specialized mathematical or computational skills have already been successfully applied in many fields of science (see for example Woodward 2012 and Chen et al. 2019). Our work provides a similar framework in the field of radiocarbon dating.

We start with a short introduction to CHRONOLOG (Section 2), followed by instructions for building OxCal Bayesian models with this tool (Section 3). We then present technical details on the conversion rules from CHRONOLOG to OxCal (Section 4), and three case-studies (Section 5). Readers not interested in OxCal technicalities can skip Section 4.

---

*Corresponding author. Email: eythan.levy@gmail.com

2. CHRONOLOG

CHRONOLOG is a software tool[1] for computer-assisted chronological research (Levy et al. 2021), available online at no charge at chrono.ulb.be. The goal of the tool is to build complex chronological models, check their consistency, and compute optimal bounds on start dates, end dates and durations. We call such models "chronological networks" (Levy et al. 2021: 1), as they focus on the interconnected aspect of the chronological data (see below). Note that CHRONOLOG is not a statistical tool, but rather uses a deterministic approach, based on graph algorithms (for mathematical details, see Levy et al. 2021: 11–16; Geeraerts et al. 2017). We first describe the CHRONOLOG data model, then its main functionalities.

2.1. Data Model

The CHRONOLOG data model rests on three types of objects: time-periods, sequences, and synchronisms.

Time-periods. A *time-period* is a contiguous interval of time, characterized by a start date, an end date, and a duration, expressed in years. These dates and durations are represented by ranges, noted [*x*, *y*]. A date between 1200 and 1300 CE, for example, is thus noted [1200, 1300]. Ranges can also feature unknown values, represented by question marks. An unknown date is thus noted as [?, ?], a *terminus post quem* of 1300 CE as [1300, ?], and a *terminus ante quem* of 1300 CE as [?, 1300]. An *exact date* of 1300 CE is noted as [1300, 1300]. The same notation is also used for durations (e.g., [5, 10] years for a duration of between 5 and 10 years). Figure 1 presents an example of a CHRONOLOG time-period starting after 1200 CE, ending before 1300 CE, and lasting 30 to 60 years.

Input bounds vs. computed bounds. A CHRONOLOG time-period features *input bounds* and *computed bounds*. The input bounds are the ranges chosen by the user and appear on the left side of the time-period (Figure 1). The *computed bounds* appear on the right side of the time-period and are automatically computed by CHRONOLOG (Levy et al. 2021: 6–7). They provide the tightest possible ranges that can be obtained by combining all the input data. In the example of Figure 1, combining the dating bounds (e.g., 1200, 1300) and the minimum duration (e.g., 30 years) yields a tightened range of [1200, 1270] for the start of the time-period and [1230, 1300] for the end of the time-period.

Sequences. A *sequence* is a set of time-periods following each other with no gaps: the end date of a time-period equals the start date of the next time-period in the sequence (cf. OxCal's "contiguous model," Bronk Ramsey 2009a: 348–349). Each time-period is contained within a sequence. Figure 2 presents an example with two sequences. The time-periods of a sequence are simply stacked on top of each other, with time flowing from above to below[2].

Synchronisms. A *synchronism* is a chronological relation between two time-periods. The Figure 2 example contains two synchronisms, represented by arrows bearing the name of the synchronism. The first one expresses that Stratum 2 starts during the reign of King Albert, and the second one that Stratum 1 ends during the reign of King Baldwin. A list of the main types of CHRONOLOG synchronisms is presented in Appendix A.

---

[1]CHRONOLOG is a Java application, distributed in *.jar* format (Java executable). It requires prior installation of the Java Runtime Environment, available for free at https://www.java.com/en/download/. Note that some browsers block the download of *.jar* files. This issue can be resolved by adjusting the browser's security settings.
[2]Note that this order corresponds to the standard graphical depiction of historical dynasties (older reigns above the younger). It is the reverse order of the standard stratigraphic order, where the latest layers are depicted above the earlier ones.
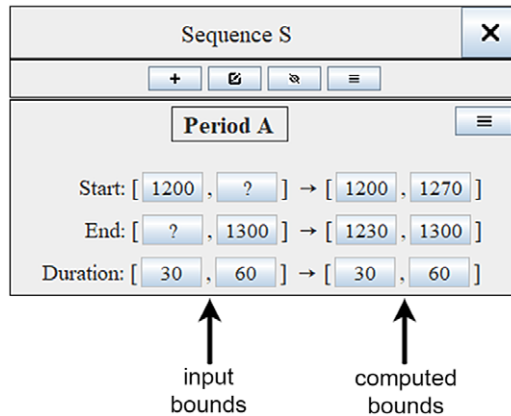
Figure 1   Graphical syntax of a CHRONOLOG time-period. The time-period has an earliest start of 1200, a latest end of 1300 and lasts 30 to 60 years. These data are shown on the left side of the time-period ("input bounds"). The right side of the time-period displays the "computed bounds," i.e., the tightest possible bounds computed by CHRONOLOG.
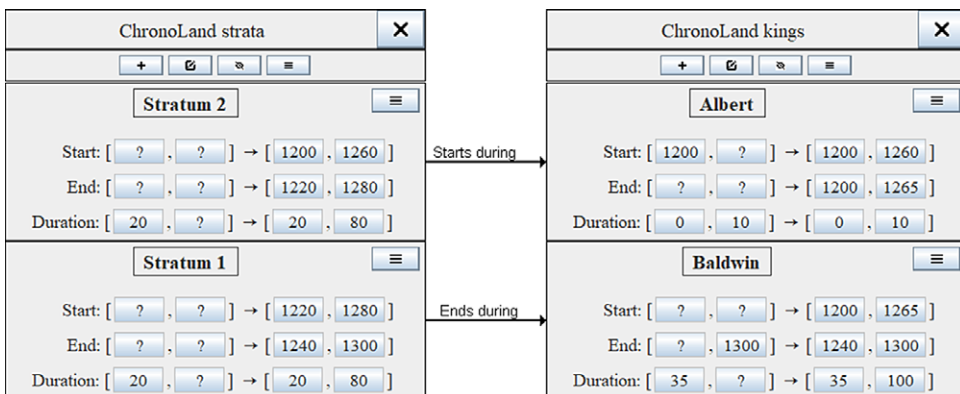

Figure 2   The ChronoLog graphical syntax. The model shows two kings and two strata. The strata have a duration of at least 20 years, King Albert started reigning after 1200 CE and reigned at most 10 years, and King Baldwin died before 1300 and reigned at least 35 years. In addition, two synchronisms connect a king to a stratum. Input bounds are displayed on the left side of the time-period, and tightened bounds on its right side.

Figure 2 presents a full CHRONOLOG model, with two archaeological strata (Stratum 2, preceding Stratum 1) and two kings (Albert, preceding Baldwin). The strata last at least 20 years each: Albert's reign starts after 1200 CE and lasts at most 10 years. Baldwin dies before 1300 and reigns at least 35 years. These data, combined with the two synchronisms, allow for a significant tightening of the ranges, as shown in the right column of each time-period.

2.2. Main CHRONOLOG Functionalities

Input bounds (for dates and durations) can easily be set/modified by clicking on the bound button inside the time-period, and synchronisms can be added by simply linking two time-periods with the mouse.
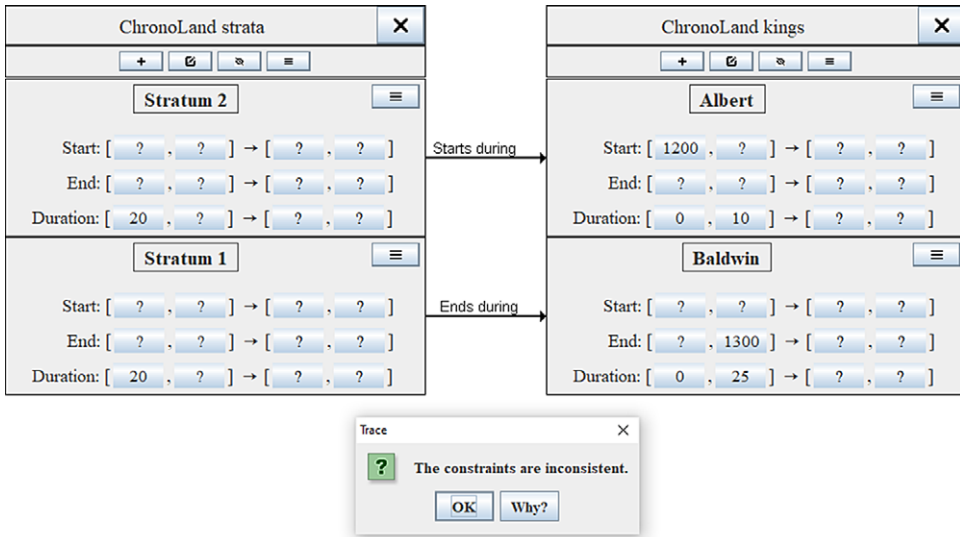
Figure 3   Same model as previously (Figure 2), but with a 25 years maximum duration for Baldwin (instead of minimum 35 years). The model is inconsistent because the resulting 35 years maximum duration of the dynasty (10+25 years) is too short to contain the 40 years minimum duration of the two strata (20+20 years). The inconsistency is automatically detected and by CHRONOLOG.

The first functionality offered by CHRONOLOG is a *consistency check*: when one enters a new bound or synchronism into a CHRONOLOG model, the system automatically checks if the data are consistent or contradictory. In the latter case, it reports the cause of the contradiction. Spotting such contradictions is not straightforward. Figure 3 shows a variant of the previous model where Baldwin is awarded a maximum duration of 25 years (instead of a minimum duration of 35 years). The resulting model is not consistent, as the new maximum duration contradicts the other constraints. Indeed, we now have 35 years maximum for the dynasty, and at least 40 years for the two strata. This creates an impossibility, since the two strata (at least 40 years) are supposedly included in the timespan of the dynasty (at most 35 years), through the two synchronisms. CHRONOLOG reports the presence of such inconsistencies (see the small popup window in Figure 3). It also reports the cause of the inconsistency (by clicking on the popup window's "Why?" button).

The second functionality is called *tightening* (see Levy et al. 2021: 6–7). It consists in computing, for each time-period, the tightest possible range for each start date, end date and duration. These computed bounds appear on the right side of each time-period, after the input bounds. The procedure for computing these bounds has been described in details elsewhere (Levy et al. 2021: 11–15). In short, the CHRONOLOG model is encoded as a large graph, where nodes represent boundaries and edges represent delays between these boundaries. The tightened bounds for each boundary are obtained by propagating the date priors along this network using shortest-path graph algorithms. This approach ensures the obtention of the tightest possible bounds derivable from the set of priors (see Geeraerts et al. 2017 and Levy et al. 2021: 11–15).

The consistency check and tightening operations run fast and are performed on-the-fly. Full details on the usage and internals of CHRONOLOG can be found in Levy et al. 2021, and online at chrono.ulb.be.
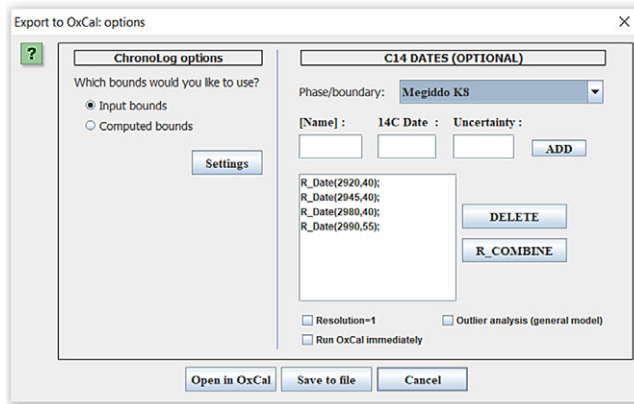
Figure 4 CHRONOLOG's "OxCal export" window.

3. BUILDING BAYESIAN OXCAL SCRIPTS WITH CHRONOLOG

3.1. User Manual

CHRONOLOG allows users to add radiocarbon dates in selected time-periods of a model, and to convert the whole model to an OxCal script. This is done in the following way:

1. Launch the CHRONOLOG "Export to OxCal" dialog (Figure 4) by using "File→Export to OxCal" or by clicking on the "OxCal export" button (atom icon) in the CHRONOLOG toolbar. Alternatively, one can also export a single sequence to OxCal by clicking on the "OxCal export" button in the sequence's toolbar.
2. Encode the (uncalibrated) radiocarbon dates (under "C14 Dates"). Each radiocarbon date is associated with a CHRONOLOG time-period (translated to an OxCal *phase*) or with a start/ end date (OxCal *boundary*[3]). These are selected by the user in the "Phase/boundary" combo box. The input of ^{14}C dates follows the classical OxCal scheme: optional name, ^{14}C date, uncertainty.
3. Dates relating to the same chronological event can be combined by selecting them with the mouse while holding the Ctrl key, then clicking the "R_Combine" button.
4. Additional options:
   a. Input bounds vs. computed (tightened) bounds: choose (under "ChronoLog options") whether the Bayesian priors of the OxCal script will be CHRONOLOG's input bounds or the tight computed bounds. The former option is the simplest and is the default used in this paper (see Section 6 for more details).
   b. Default resolution vs. 1 year-resolution: a checkbox allows use of a resolution of 1 year instead of the default resolution provided by the calibration curve.
   c. Outlier analysis: a checkbox allows automatic insertion of code for outlier analysis, using OxCal's "general model" (Bronk Ramsey 2009b: 1028).

---

[3]For an example of inclusion of ^{14}C determinations inside a boundary, see Garfinkel et al. 2019 (supplementary material).

    d. Export to file vs. opening in OxCal: either save the OxCal script to a file (button "Save to file") or open it directly online on the OxCal website[4] (button "Open in OxCal"). In the latter case, the "Run OxCal immediately" checkbox also runs the script immediately on the OxCal site.

3.2. Advantages

This approach has several advantages. First, it automatically detects possible contradictions among the encoded priors. As noted above, manually detecting such contradictions is not an easy task. Our approach allows an early detection of any inconsistency, before running the OxCal script, resulting in potentially significant time saving.

Second, the tightening algorithm used by CHRONOLOG guarantees the tightest possible ranges derivable from the set of priors (Levy et al. 2021: 6–7, 12–14). CHRONOLOG allows users to build OxCal models based on these tightened ranges rather than the input priors (see item 4a above). The inclusion of these tight ranges might help (or accelerate) the execution of OxCal in selected cases, where OxCal's MCMC algorithm fails to start (or starts too slowly) because of difficulties in finding a first reasonable solution (cf. Bronk Ramsey 2009a: 357–358).

The tight bounds can also be useful when one wants to focus on a single stratigraphic sequence of a complex CHRONOLOG model. In that case, the model can be compacted by exporting that stratigraphic sequence only and using the tight bounds. This approach builds a reduced OxCal model that features only one sequence but enriches it with priors deriving from the complete CHRONOLOG model.

The main advantage of our approach is that it allows users with no knowledge of the OxCal language to build Bayesian models by themselves. CHRONOLOG allows them to automatically launch the execution of the script on the OxCal website, and to simply wait for the OxCal report table which displays the confidence intervals for each boundary. Beginners in OxCal might also find that their ChronoLog-generated OxCal models more accurately reflect archaeological thinking, as the OxCal commands in these scripts directly translate the conceptual objects represented in the CHRONOLOG model.

For more experienced users, our approach allows quick building of an initial model skeleton, which they can then easily adjust by enriching or modifying the script directly in OxCal, thus saving significant development time. Such users might also find that using CHRONOLOG-generated models reduces the risk of inadvertent coding errors. It is indeed easy to unwittingly create an OxCal model with a bug that may never be discovered. Having CHRONOLOG generate the OxCal model skeleton is thus an advantage in this matter.

We do not claim that CHRONOLOG is meant to replace direct modeling with OxCal. First, OxCal experts might prefer to use the original tool, in order to take full advantage of all the subtleties and advanced modeling options offered by the OxCal language. Furthermore, CHRONOLOG does not yet implement the full array of existing OxCal instructions, but rather focuses on the most common ones. Finally, CHRONOLOG is particularly useful for cases where the set of prior constraints forms a large part of the model, for example in cases of synchronisms and duration bounds derived from historical sources. In such cases, our approach provides an easy way to convert the CHRONOLOG model to a Bayesian

---

[4]The latter option (open directly in OxCal) has a limit on the size of the script, imposed by the web server. For large scripts, the "Save to file" option is preferred.
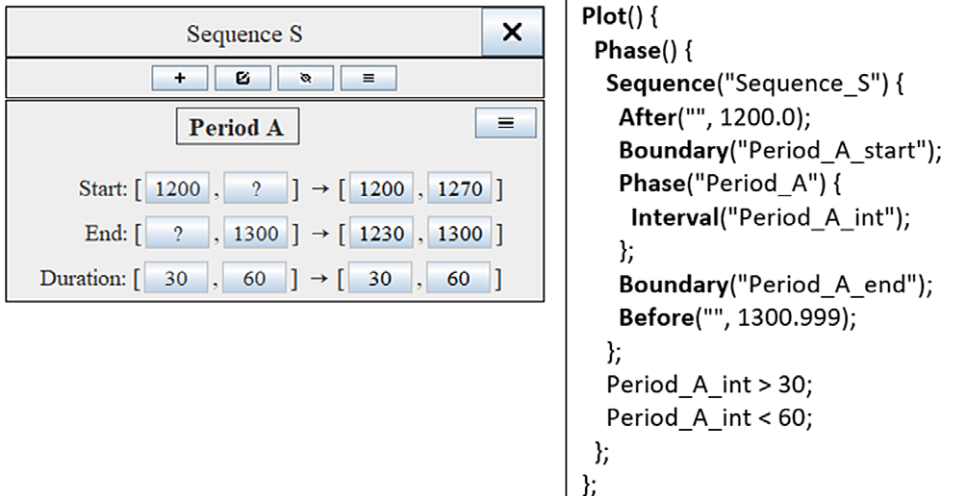
Figure 5   A CHRONOLOG time-period (with date and duration bounds), and the automatically generated OxCal script (using input bounds, rather than computed bounds, and omitting radiocarbon dates). Note: a ChronoLog earliest date of X is modeled in OxCal as After("", X.0), with X.0 meaning the start of year X. A ChronoLog latest date of X is modeled in OxCal as Before("", X.999), to represent the end of year X.

OxCal script with the same constraints. For other cases however, like prehistoric sites with a few phases, or environmental sequences, CHRONOLOG might not always provide an easier way to get started.

Note that our CHRONOLOG-to-OxCal conversion tool has been tested with OxCal version 4.4 (https://c14.arch.ox.ac.uk/oxcal/OxCal.html) and might not be fully compatible with other versions of OxCal.

4. CHRONOLOG TO OXCAL CONVERSION RULES

This section describes how CHRONOLOG models are translated into OxCal code. For the sake of simplicity, all the examples use the CHRONOLOG input bounds (rather than the computed ones). Readers not interested in the OxCal technicalities can skip this section.

4.1. Time-periods

Each CHRONOLOG time-period is translated to an OxCal *Phase*. Lower and upper bounds on start/end dates are modeled with the OxCal *After( )* and *Before( )* instructions, respectively. Exact dates are encoded directly into the boundary (e.g., *Boundary("Albert_start", CE(1200))*). Duration bounds are encoded using an *Interval( )* statement inside the Phase and an interval constraint after the Phase. An exact duration is encoded using a boundary constraint (*boundary_end = boundary_start + duration*). Figure 5 shows an example.

4.2. Sequences

CHRONOLOG sequences are translated as *contiguous* OxCal multiphase sequences i.e., sequences featuring one OxCal boundary between each pair of consecutive phases (Bronk

```
Plot() {
 Phase() {
  Sequence("Sequence S") {
   Boundary("Period_A_start", CE(1200));
   Phase("Period_A") {
    Interval("Period_A_int");
   };
   Boundary("Period_B_start");
   Phase("Period_B") {
   };
   Boundary("=Period_B_end");
   Before("", 1300.999);
  };
  Period_A_int > 30;
  Period_A_int < 50;
  Period_B_end = Period_B_start + 20;
 };
};
```

Figure 6   Converting a ChronoLog sequence to OxCal code (using input bounds and omitting radiocarbon dates).

Ramsey 2009a: 348–349).[5] Such sequences exactly model the semantics of CHRONOLOG sequences (see Section 2.2 above). An example of this is seen in Figure 6. In addition, if the initial phase of the sequence does not feature a ^{14}C date or an earliest start, CHRONOLOG automatically assigns it a default earliest start (currently set to 50,000 BCE). Similarly, if the last phase of the sequence has no ^{14}C date nor a latest end, CHRONOLOG assigns it a default latest end of 1950 CE. The default bounds are meant to ease the OxCal computation process. They can be modified in the CHRONOLOG "OxCal export" window, by clicking on the "Settings" button.

4.3. Synchronisms

CHRONOLOG synchronisms are translated as OxCal boundary constraints inserted at the bottom of the script. All the types of CHRONOLOG synchronisms can be translated to OxCal, as all of them are expressed using simple relations ($\leq$, $\geq$, $=$) between boundaries (see Figures 16 and 17). Figure 7 provides an example using the CHRONOLOG *contemporaneity* synchronism, which expresses that two time-periods share some amount of time (see Figure 17).

Appendix B presents the complete conversion rules from CHRONOLOG to OxCal.

5. CASE STUDIES

We present three case-studies taken from recent radiocarbon publications. The CHRONOLOG files of the case-studies are available online on the CHRONOLOG website at chrono.ulb.be/download/. These case-studies are presented as proofs of concept, to show how use of our method enables users to reach similar results in a straightforward, effortless manner. All the scripts were executed with OxCal version 4.4. The complete OxCal scripts are provided in Appendix C.

---

[5]To model a sequence with potential gaps between its phases (OxCal's "Sequential multiphase model," cf. Bronk Ramsey 2009a: 348–349), gap time-periods must be explicitly added between the CHRONOLOG time-periods.
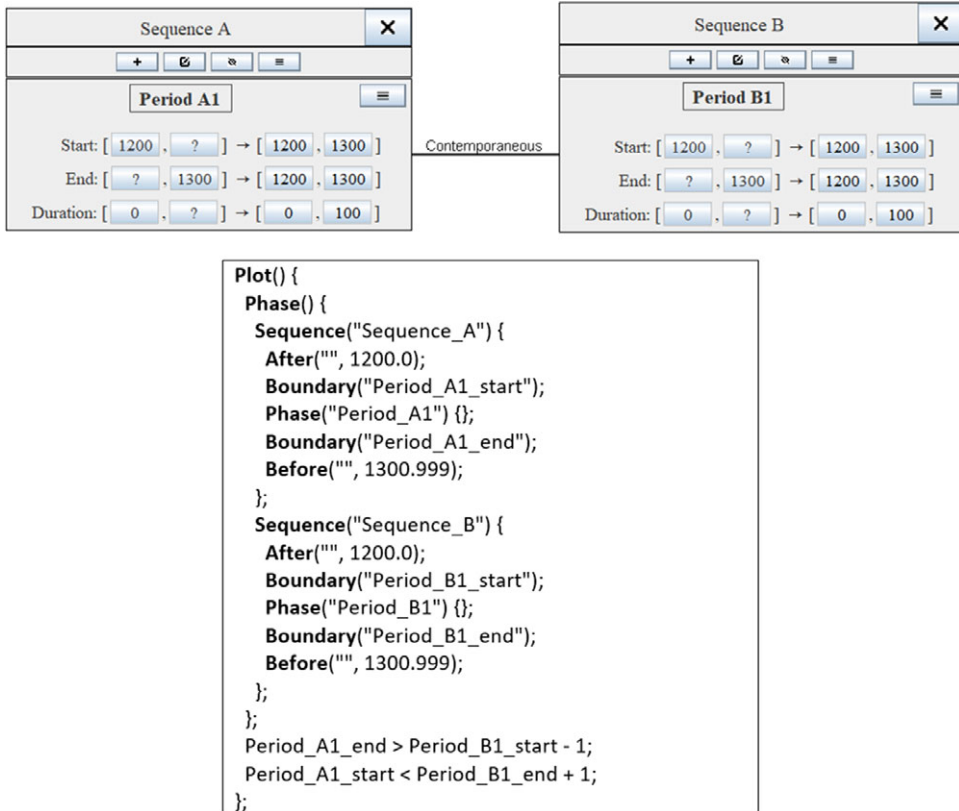
```
Plot() {
 Phase() {
  Sequence("Sequence_A") {
   After("", 1200.0);
   Boundary("Period_A1_start");
   Phase("Period_A1") {};
   Boundary("Period_A1_end");
   Before("", 1300.999);
  };
  Sequence("Sequence_B") {
   After("", 1200.0);
   Boundary("Period_B1_start");
   Phase("Period_B1") {};
   Boundary("Period_B1_end");
   Before("", 1300.999);
  };
 };
 Period_A1_end > Period_B1_start - 1;
 Period_A1_start < Period_B1_end + 1;
};
```

Figure 7 Example of converting a contemporaneity synchronism to OxCal. Radiocarbon dates are omitted for the sake of conciseness. The OxCal constraints corresponding to the CHRONOLOG synchronism are shown in bold. The +1 and –1 have been added to emulate CHRONOLOG's ≥ and ≤ operators using OxCal's ">" and "<" operators.

5.1. Egyptian 18th Dynasty (Bronk Ramsey et al. 2010; Quiles et al. 2013)

Our first example concerns the chronology of the second half of the Egyptian 18th dynasty (Kings Amenhotep II to Horemheb). Several recent studies have reconstructed the ancient Egyptian chronology based on Bayesian radiocarbon models (Bronk Ramsey et al. 2010; Quiles et al. 2013; Shortland and Bronk Ramsey 2013). These models include radiocarbon samples from contexts associated with known kings, as well as Bayesian priors that model the sequence of kings and provide estimates of the durations of their reigns.

We present a simple such model, with data gathered from previous studies (Bronk Ramsey et al. 2010; Quiles et al. 2013). Our model features eight kings, five of whom are associated with contemporary radiocarbon samples (the determinations are from Bronk Ramsey et al 2010). All of the kings are given duration ranges, adapted from Quiles et al. 2013. In addition, the model features radiocarbon determinations from the tomb of Sennefer (Quiles et al. 2013), a tomb dated between the start of Tutankhamen's reign and the start of Horemheb's reign. We also add a *terminus ante quem* (TAQ) of 1200 BCE for the end of the dynasty. This TAQ is very conservative, as it is ca. 100 years after the accepted historical date for the dynasty's end. Its inclusion in the model is necessary in order to help OxCal start the Monte Carlo Markov Chain (MCMC) algorithm because
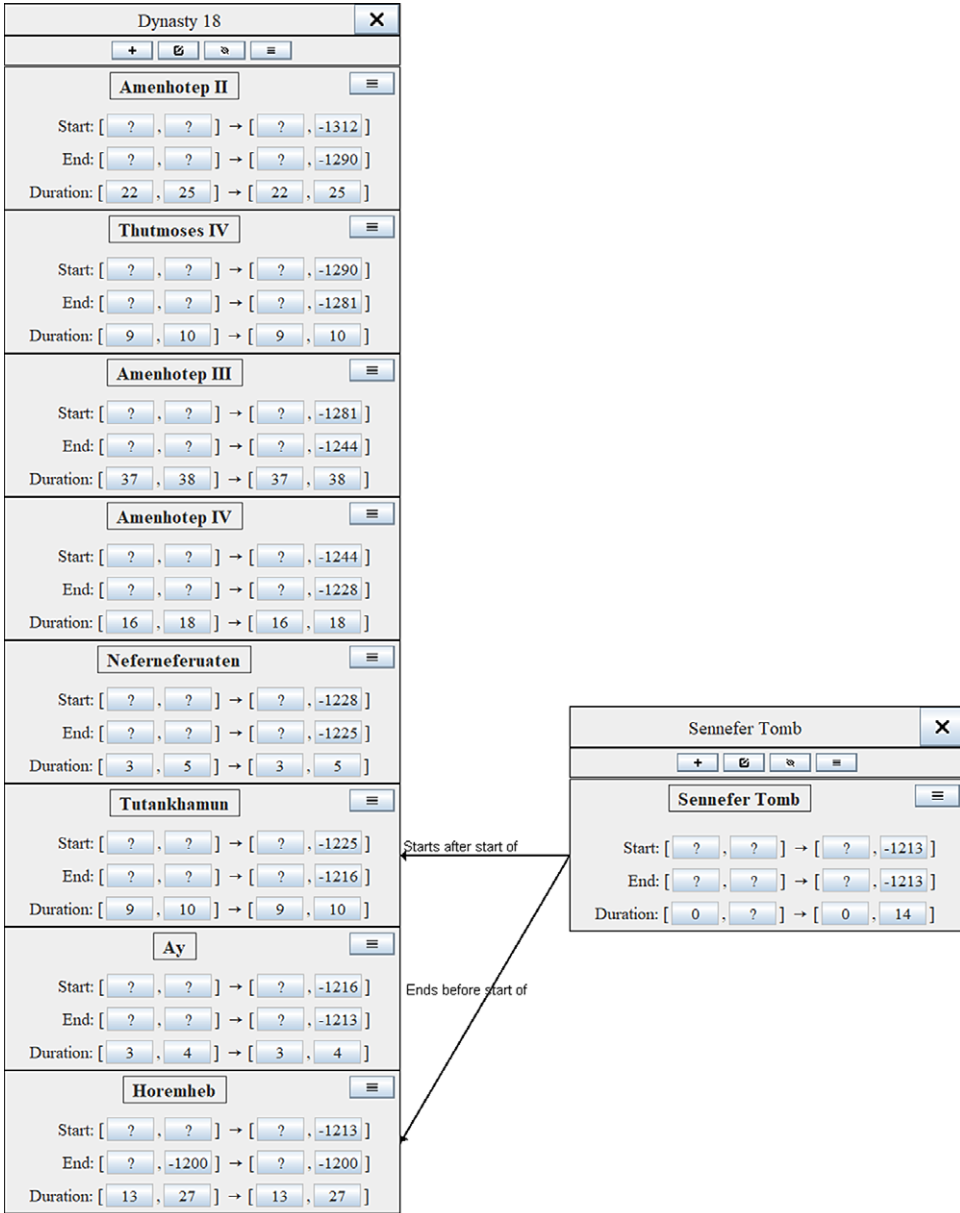
Figure 8   CHRONOLOG model for the second half of the Egyptian 18th dynasty. All reigns feature a duration estimate adapted from Quiles et al. (2013). The Sennefer tomb is dated to between the start of Tutankhamen and the start of Horemheb. The model includes radiocarbon determinations (from Bronk Ramsey et al. 2010) in all reigns, except Neferneferuaten, Ay and Horemheb. Further radiocarbon determinations are included in the tomb of Sennefer (from Quiles et al. 2013). See Appendix C for full details on the model.

our model features no radiocarbon determinations for the last two reigns (see Bronk Ramsey 2009a: 357–358).

Figure 8 shows the CHRONOLOG model. The left side displays the sequence of kings, with their duration ranges and the TAQ. The tomb of Sennefer appears on the right side, with two

```
Plot() {
   Phase() {
      Sequence("Dynasty_18") {
         Boundary("Amenhotep_II_start");
         Phase("Amenhotep_II") {
           Interval("Amenhotep_II_int");
           /* C14 DATES HERE */
         };
         Boundary("Thutmoses_IV_start");
         Phase("Thutmoses_IV") {
            Interval("Thutmoses_IV_int");
            /* C14 DATES HERE */
         };
         :
         :  /* OTHER REIGNS HERE */
         Boundary("Horemheb_start");
         Phase("Horemheb"){
            Interval("Horemheb_int");
         };
         Boundary("Horemheb_end");
         Before("", -1200.999);
      };
      Amenhotep_II_int > 22;
      Amenhotep_II_int < 25;
      Thutmoses_IV_int > 9;
      Thutmoses_IV_int < 10;
      :
      :  /* OTHER REIGN DURATIONS HERE*/   .
      Horemheb_int > 13;
      Horemheb_int < 27;

      Sequence("Sennefer_Tomb") {
         Boundary("Sennefer_Tomb_start");
         Phase("Sennefer_Tomb") {
            /* C14 DATES HERE */
         };
         Boundary("Sennefer_Tomb_end");
      };
   };
   Sennefer_Tomb_start > Tutankhamun_start - 1;
   Sennefer_Tomb_end < Horemheb_start + 1;
};
```

Figure 9   Egyptian 18th dynasty: sketch of the OxCal script generated by CHRONOLOG. The sketch omits radiocarbon determinations and several reigns, for the sake of conciseness (see Appendix C for the full script).

synchronisms that place it between the starts of the reign of Tutankhamun and of Horemheb. The graphical syntax of the CHRONOLOG model is simple and self-explanatory. Figure 9 presents a sketch of the OxCal script generated by CHRONOLOG, based on the input bounds (see Appendix C for the full script). The script is fairly classical, with two sequences (kings, Sennefer), and two synchronisms at the bottom. It was generated using a one-year resolution and the "general model" of outlier analysis (see Section 3.1 above), in order to

Table 1 Results of the OxCal script generated by CHRONOLOG, compared to those of Bronk Ramsey et al. (2010).

Accession date	Our model		Bronk Ramsey et al. (2010)	
	68% range	95% range	68% range	95% range
Amenhotep II	1426–1418	1430–1415	1441–1431	1445–1423
Thutmose IV	1402–1395	1406–1392	1414–1403	1418–1396
Amenhotep III	1393–1385	1396–1382	1404–1393	1408–1386
Amenhotep IV	1355–1348	1359–1345	1365–1355	1370–1348
Neferneferuaten	1338–1331	1342–1328	1351–1340	1356–1333
Tutankhamun	1334–1327	1338–1324	1349–1338	1353–1331
Ay	1325–1318	1328–1315	1339–1329	1344–1322
Horemheb	1321–1314	1325–1311	1336–1325	1341–1318
End of dynasty	1305–1293	1309–1288	1308–1297	1313–1290

conform to the original publication (Bronk Ramsey et al. 2010). Table 1 presents the results obtained by running the script, compared to those in Bronk Ramsey et al. 2010. Note the close similarity between the two sets of results. The slight variations might stem from the following factors: (1) the latter model was much wider in time, covering the 17th to 21st dynasties, (2) the reign duration constraints are slightly different,[6] (3) the latter model used a regional offset of 19 ±5 radiocarbon years (Bronk Ramsey et al. 2010: 1555) absent from our model.

This first example was intended to demonstrate the ease of creating a historical sequence (with duration bounds and TPQs/TAQs) and simple synchronization features (tomb of Sennefer) using CHRONOLOG. Furthermore, the CHRONOLOG graphical depiction of the model makes it straightforward to read and to modify. Note that the script has been entirely generated by CHRONOLOG (including the one-year resolution parameter and the outlier analysis code) and can be run by OxCal without further modification[7]. Such is also the case for the scripts of the following two case studies.

5.2. Late Helladic to Proto-Geometric Aegean Chronology (Fantalkin et al. 2015)

We now illustrate the creation of more complex models, that feature archaeological strata contained within cultural phases. Our model is adapted from Fantalkin et al. (2015). Their study featured a carbon-based chronology for Aegean cultural phases, from the late Late Helladic IIIB to the Middle Geometric II period, primarily based on Aegean ceramic imports to the Levant. They built a large Bayesian model, featuring 19 archaeological strata contained within 12 cultural phases, as well as *termini post* and *ante quem* (Figure 10). For simplicity, we limited the CHRONOLOG model presented here to the first six cultural phases (LH IIIB to Early Proto-Geometric; see Figure 11).

Fantalkin et al.'s model used nested OxCal phases for modeling inclusion of strata within cultural phases, as is customary in OxCal modeling. However, CHRONOLOG does not use

---

[6]We treated all duration ranges as uniform, rather than using more complex models featuring Student's distribution (Bronk Ramsey et al. 2010: 1555).

[7]In our three case-studies, the only code manually added to the models is a line for changing the calibration curve. In order to ease comparison with the published results, we used the same calibration curve as in the original publications (IntCal04 for the first and last case study, IntCal13 for the second case study). Note that the full scripts given in Appendix C are the original OxCal scripts generated by CHRONOLOG, without change of the calibration curve.
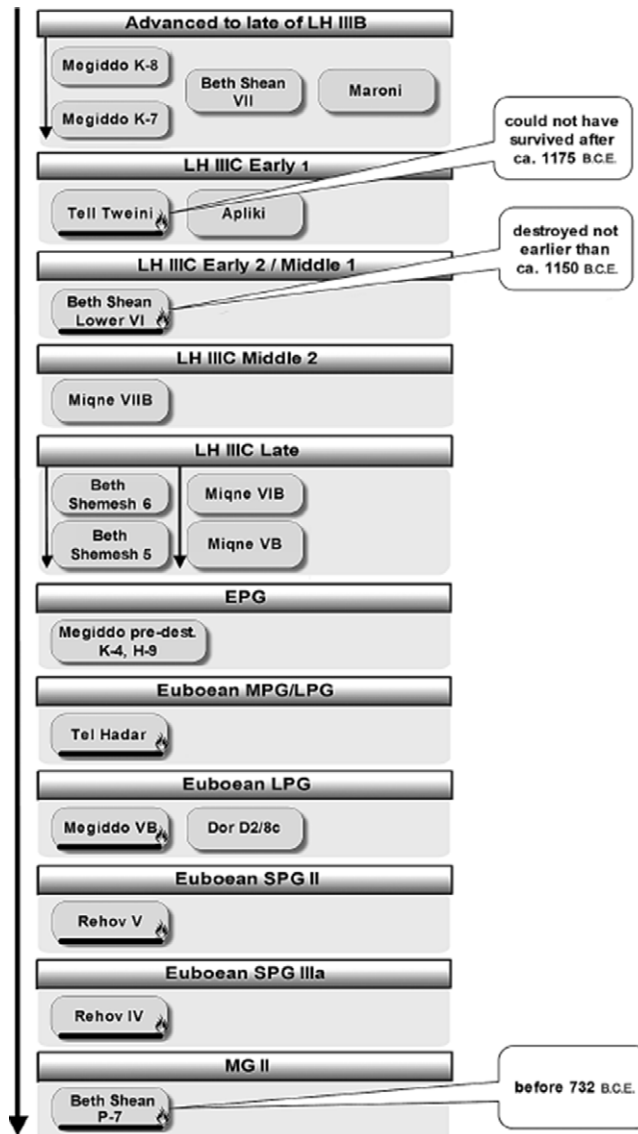
Figure 10 Bayesian model of Fantalkin et al. (2015: 30, their Fig. 2) for Aegean chronology from the late Late Helladic IIIB to the Middle Geometric II.

nesting of phases, but rather *inclusion synchronisms* (see Figure 17). The generated OxCal script therefore features one (carbon-less) sequence of cultural phases, and several independent sequences for the strata. Inclusion of strata within a cultural phase is represented by the chronological constraints located at the bottom of the script (Figure 12).

Table 2 compares the results of the script with those of the original paper. The two sets of results show good general agreement, but also some discrepancies, which can be explained by the following considerations:
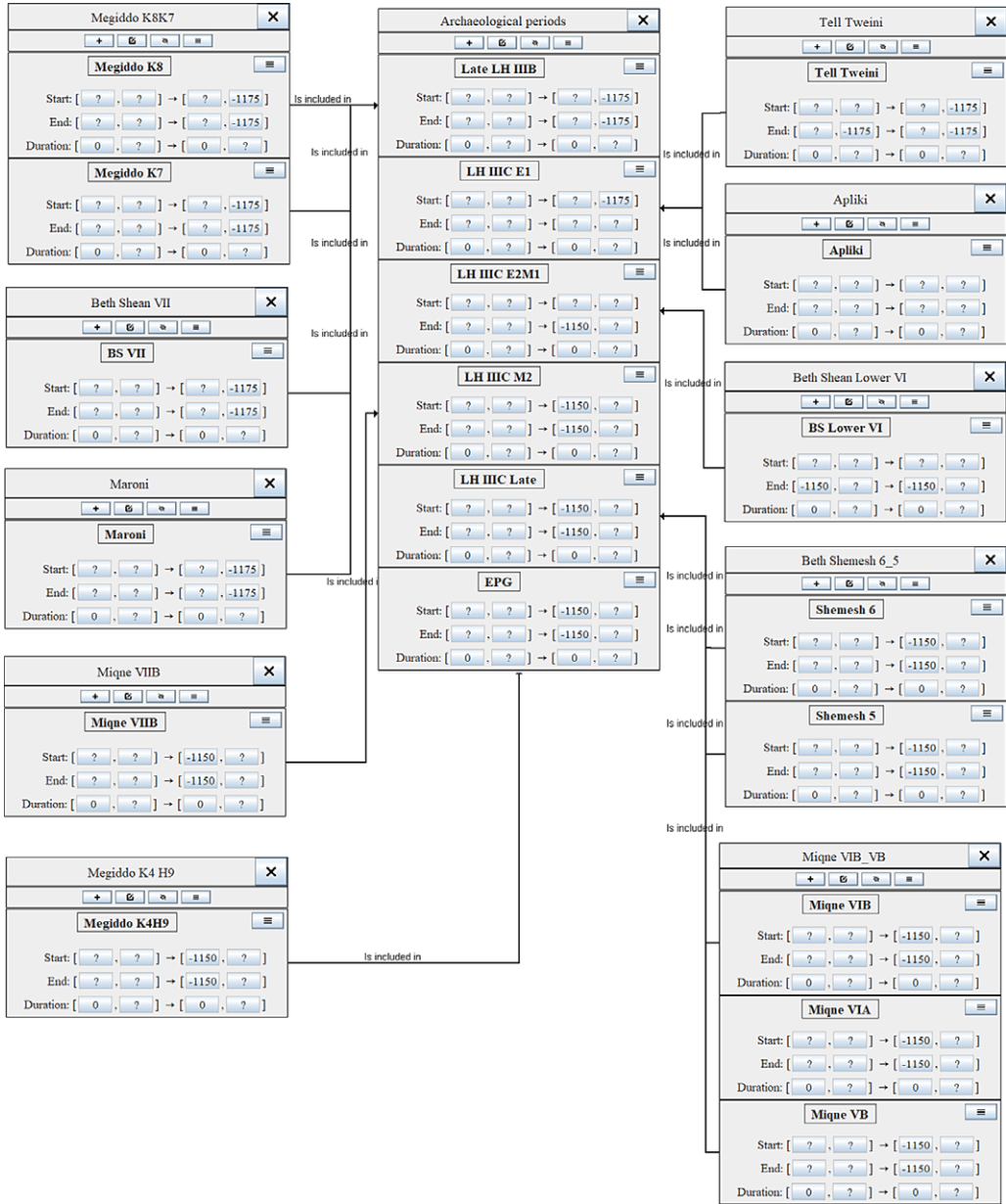
Figure 11 CHRONOLOG model for Aegean chronology from the late Late Helladic IIIB to the Early Proto-Geometric (adapted from Fantalkin et al. 2015). All the synchronisms are of type "A is included in B".

1. Our model is smaller than the original one: we truncated the model at the end of Early Proto-Geometric, to ease the presentation.

2. Our model uses synchronisms instead of nested phases[8].

---

[8] Another consequence of using inclusion synchronisms rather than nested phases lies in different results for the very first and very last boundaries of a sequence. With inclusion synchronisms, OxCal will not compute a lower bound for the very first boundary, nor an upper bound for the very last boundary, but will rather use the earliest start and latest end provided by the CHRONOLOG model.

```
Plot() {
 Phase() {
  Sequence("Archaeological_periods") {
   After("", -50000);
   Boundary("Late_LH_IIIB_start");
   Phase("Late_LH_IIIB") { };
   Boundary("LH_IIIC_E1_start");
   Phase("LH_IIIC_E1") { };
   Boundary("LH_IIIC_E2M1_start");
   Phase("LH_IIIC_E2M1") { };
   Boundary("LH_IIIC_M2_start");
   Phase("LH_IIIC_M2") { };
   Boundary("LH_IIIC_Late_start");
   Phase("LH_IIIC_Late") { };
   Boundary("EPG_start");
   Phase("EPG") { };
   Boundary("EPG_end");
   Before("", 1950);
  };
  Sequence("Megiddo_K8K7") {
   Boundary("Megiddo_K8_start");
   Phase("Megiddo_K8") {
    /* C14 DATES HERE */
   };
   Boundary("Megiddo_K7_start");
   Phase("Megiddo_K7") {
    /* C14 DATES HERE */
   };
   Boundary("Megiddo_K7_end");
  };

    ⋮  /* OTHER STRATA HERE */

  Sequence("Megiddo_K4_H9") {
   Boundary("Megiddo_K4H9_start");
   Phase("Megiddo_K4H9") {
    /* C14 DATES HERE */
   };
   Boundary("Megiddo_K4H9_end");
  };
 };
 Megiddo_K8_start > Late_LH_IIIB_start - 1;
 Megiddo_K7_start < LH_IIIC_E1_start + 1;
 Megiddo_K7_start > Late_LH_IIIB_start - 1;
 Megiddo_K7_end < LH_IIIC_E1_start + 1;

    ⋮  /* OTHER INCLUSION SYNCHRONISMS HERE */

 Megiddo_K4H9_start > EPG_start - 1;
 Megiddo_K4H9_end < EPG_end + 1;
};
```

Figure 12   Sketch of the OxCal script automatically generated by CHRONOLOG for the Aegean chronology case-study (see Appendix C for the full script).

3. We used a model of sequences that is different from theirs. We use OxCal's "contiguous model" (Bronk Ramsey 2009a: 348–349), which features one boundary between each pair of consecutive phases. Fantalkin et al. rather used the "sequential" model (*ibid.*), which features two boundaries between consecutive phases, to allow a gap between them. The dates they provide for each transition (Table 2 below) are obtained using the

Table 2   Results of the OxCal script generated by CHRONOLOG, compared to those of Fantalkin et al. (2015: 31, their Table 2).

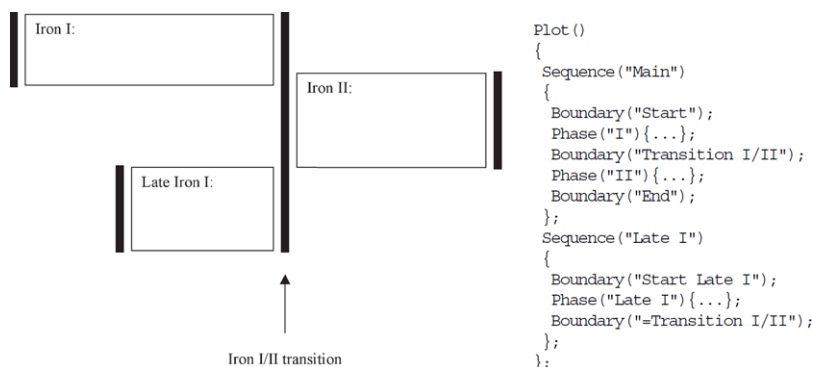| Start boundary	Our model		Fantalkin et al. (2015)	
	68% range	95% range	68% range	95% range
LH III C Early 1	1198–1182	1205–1177	1205–1176	1209–1175
LH III C Early 2/Middle 1	1171–1140	1180–1125	1188–1132	1199–1120
LH IIIC Middle 2	1135–1100	1147–1082	1145–1083	1150–1060
LH IIIC Late	1095–1055	1114–1042	1098–1035	1120–1026
Early Proto-Geometric	1040–1015	1064–1003	1049–1018	1081–1010

Figure 13   Bayesian model for the Iron I/Iron II transition in the southern Levant (Bronk Ramsey 2009a: 351, their Figure 7, adapted from Mazar and Bronk Ramsey 2008, their Figure 2).

start of the confidence interval of the first boundary, and the end of the confidence interval of the second boundary (Fantalkin et al. 2015: 35–36).

5.3. Iron I/Iron II Transition in the Southern Levant (Mazar and Bronk Ramsey 2008)

Our final case-study concerns a radiocarbon model of the Iron I/Iron II transition in the southern Levant (Mazar and Bronk Ramsey 2008, Model C3). This model uses an Iron I phase, an Iron II phase, and an additional "Late Iron I" phase whose end is synchronized with the end of Iron I (Figure 13). Twenty-two additional phases representing archaeological strata bearing short-lived ^{14}C samples are nested within these three cultural phases. A CHRONOLOG representation of this model is given in Figure 14. The model uses two CHRONOLOG sequences: one for Iron I and Iron II, and another one for "Late Iron I." The end of Iron I and "Late Iron I" are synchronized through a "Simultaneous end" synchronism. Twenty-two additional time-periods (not shown in Figure 14 through lack of space) represent the strata and are connected to the three main time-periods using "inclusion synchronisms," as in the previous model. A sketch of the OxCal script generated by CHRONOLOG is given in Figure 15 (see Appendix C for the full script). The script contains the same radiocarbon determinations as in Mazar and Bronk Ramsey's Model C3, except for two samples (lab nos. HM3 and QS6), which had to be removed in order to reach an agreement index above 60%. The script uses a one-year resolution in order to conform to
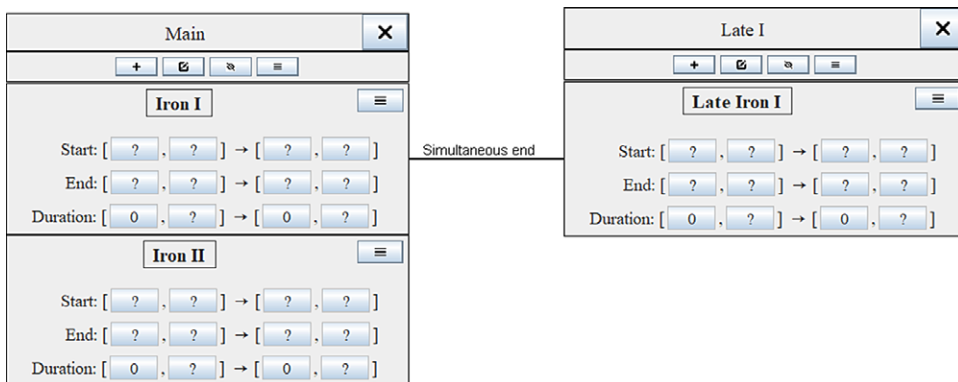
Figure 14 CHRONOLOG representation of the Mazar and Bronk Ramsey model shown in Figure 13 (partial view, hiding the strata included in each cultural phase, see Appendix C for the full CHRONOLOG model).

```
Plot() {
 Phase() {
  Sequence("Main") {
   Boundary("Iron_I_start");
   Phase("Iron_I") { ... };
   Boundary("Iron_II_start");
   Phase("Iron_II") { ... };
   Boundary("Iron_II_end");
  };
  Sequence("Late_I") {
   Boundary("Late_Iron_I_start");
   Phase("Late_Iron_I") { ... };
   Boundary("=Late_Iron_I_end");
  };
 };
 Late_Iron_I_end = Iron_II_start + 0;
};
```

Figure 15 OxCal script generated by CHRONOLOG on the basis of the model of Figure 14 (omitting ^{14}C dates, see Appendix C for full model).

the original publication. Table 3 compares the results of our script for the Iron I/Iron II transition to those of Mazar and Bronk Ramsey. The results are very close: at 68% we have 949–928 vs. their 948–919, and at 95% we have 958–917 vs. their 963–917. Here, as before, the slight differences can easily be explained by differences in modeling.

Finally, we would like to illustrate how easily CHRONOLOG enables us to enrich a model. For example, we could posit that the "late Iron I" begins after the start of Iron I, which is actually part of the very definition of "late Iron I" but was not represented in the above model. This can be done very easily, by adding one more synchronism between Iron I and "late Iron I," namely "late Iron I starts after the start of Iron I" (Figure 16). It also shows that CHRONOLOG permits having more than one synchronism between two given time-periods. This example is given here

Table 3   Results of the OxCal script generated by CHRONOLOG, compared to those of Mazar and Bronk Ramsey (2008: 173, their Figure 1, Model C3).

	Our model		Mazar and Bronk Ramsey (2008)	
	68% range	95% range	68% range	95% range
Iron I/Iron II transition	949–928	958–917	948–919	963–913

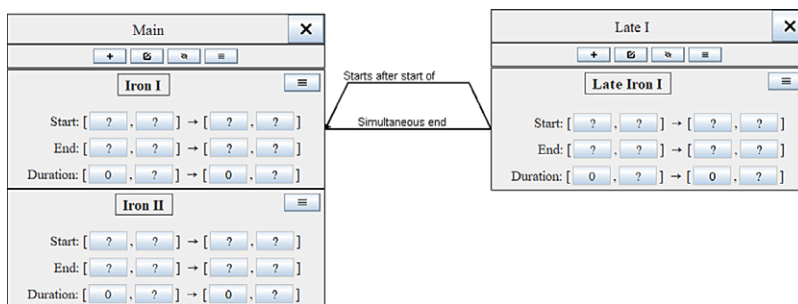Figure 16   Alternative model, imposing the additional constraint: "Late Iron I starts after the start of Iron I".

for illustrative purposes only, as the generated script gave exactly the same result for the Iron I/Iron II transition as the preceding one.

6. CONCLUSION

This paper demonstrated how to generate Bayesian models automatically using CHRONOLOG, chronological modeling software (Levy et al. 2021) freely available online at chrono.ulb.be. CHRONOLOG enables users with no computer programming skills, or with no knowledge of the OxCal language, to create complex radiocarbon Bayesian models with the help of a user-friendly interface. This approach has the potential to significantly widen the pool of current practitioners of OxCal Bayesian modeling, by making archaeologists less dependent on programmers for building models. For more experienced OxCal users, CHRONOLOG can represent a significant time saver, allowing them to generate the skeleton of their OxCal scripts automatically, which they can then fine-tune manually. In addition, CHRONOLOG's rich library of high-level synchronisms can be particularly useful for helping users build complex Bayesian models. Our three case-studies showed that our approach produces similar results to those obtained with OxCal models written manually. Yet, as often occurs in automatically generated code, the OxCal scripts generated by CHRONOLOG are sometimes slower than their manual counterparts. This is due to the former scripts using more OxCal boundaries than manual ones. Furthermore, in the case of strata included within cultural periods, our approach based on synchronisms runs slower than OxCal's classical approach of nested phases (see Section 5.2). Future developments of CHRONOLOG will focus on optimizing the generated OxCal code to address these shortcomings.

Finally, we intend to add more OxCal functionalities in future versions of CHRONOLOG, such as non-uniform phases (with features such as *Zero_Boundary*, *Tau_Boundary*, and

Sigma_Boundary), and more advanced types of outlier analysis than the currently implemented general model (cf. Bronk Ramsey 2009b).

ACKNOWLEDGMENTS

REFERENCES

Bronk Ramsey C. 2009a. Bayesian analysis of radiocarbon dates. Radiocarbon 51(1):337–360.

Bronk Ramsey C. 2009b. Dealing with outliers and offsets in radiometric datings. Radiocarbon 51(3): 1023–1045.

Bronk Ramsey C, Dee MW, Rowland JM, Higham TFG, Harris SA, Brock F, Quiles A, Wild EM, Marcus ES, Shortland AJ. 2010. Radiocarbon-based chronology for dynastic Egypt. Science 328:1554–1557.

Bronk Ramsey C, Shortland AJ, editors. 2013. Radiocarbon and the chronologies of Ancient Egypt. Oxford and Oakville: Oxbow Books.

Chen J, Zhang R, Dong X, Lin L, Zhu Y, He J, Christiani DC, Wei Y, Chen F. 2019. shinyBN: an online application for interactive Bayesian network inference and visualization. BMC Bioinformatics 20:711.

Dye TS, Buck CE. 2015. Archaeological sequence diagrams and Bayesian chronological models. Journal of Archaeological Science 63:84–93.

Fantalkin A, Finkelstein I, Piasetzky E. 2015. Late Helladic to Middle Geometric Aegean and contemporary Cypriot chronologies: a radiocarbon view from the Levant. Bulletin of the American Schools of Oriental Research 373:25–48.

Garfinkel Y, Hasel MG, Klingbeil MG, Kang H-G, Choi G, Chang S-Y, Hong S, Ganor S, Kreimerman I, Bronk Ramsey C. 2019. Lachish fortifications and state formation in the Biblical Kingdom of Judah in light of radiometric datings. Radiocarbon 61(3):695–712.

Geeraerts G, Levy E, Pluquet F. 2017. Models and algorithms for chronology. In: Schewe S, Schneider T, Wijsen J, editors. 24th International Symposium on Temporal Representation and Reasoning (TIME 20, 17). Vol. 90 of Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl: Schloss Dagstuhl–Leibniz-Zentrum Fuer Informatik. p. 13:1–13:18.

Levy E, Geeraerts G, Pluquet F, Piasetzky E, Fantalkin A. 2021. Chronological networks in archaeology: a formalised scheme. Journal of Archaeological Science 127:1–27.

Mazar A, Bronk Ramsey C. 2008. [14]C dates and the Iron Age chronology of Israel: a response. Radiocarbon 50(2):159–180.

Quiles A, Aubourg E, Berthier B, Delque-Količ E, Pierrat-Bonnefois G, Dee MW, Andreu-Lanoë G, Bronk Ramsey C, Moreau C. 2013. Bayesian modelling of an absolute chronology for Egypt's 18th Dynasty by astrophysical and radiocarbon methods. Journal of Archaeological Science 40(1):423–432.

Woodward P. 2012. Bayesian analysis made simple. An Excel GUI for WinBUGS. Boca Raton: CRC Press.

Appendix A. Main CHRONOLOG Synchronisms

Name	Image	Definition
A is contemporary with B		$end(A) \geq start(B)$ **and** $end(B) \geq start(A)$
A is included in B		$start(A) \geq start(B)$ **and** $end(A) \leq end(B)$
A includes B		$start(A) \leq start(B)$ **and** $end(B) \leq end(A)$
A overlaps with succeeding B		$start(A) \leq start(B) \leq end(A) \leq end(B)$
A overlaps with preceding B		$start(B) \leq start(A) \leq end(B) \leq end(A)$
A starts during B		$start(B) \leq start(A) \leq end(B)$
A includes start of B		$start(B) \leq start(A) \leq end(B)$
A ends during B		$start(B) \leq end(A) \leq end(B)$
A includes end of B		$start(A) \leq end(B) \leq end(A)$
Synchronous start		$start(A) = start(B)$
Synchronous end		$end(A) = end(B)$
Equality		$start(A) = start(B)$ **and** $end(A) = end(B)$
A precedes immediately B		$end(A) = start(B)$
A follows immediately B		$end(B) = start(A)$

Figure 17   The contemporaneity synchronism and special cases thereof. In the images, time flows from above to below.

Name	Image	Definition
A starts before the start of B		$start(A) \leq start(B)$
A starts after the start of B		$start(A) \geq start(B)$
A starts before the end of B	All except :	$start(B) \leq end(A)$
A starts after the end of B		$start(A) \geq end(B)$
A ends before the start of B		$ends(A) \leq start(B)$
A ends after the start of B	All except :	$end(A) \geq start(B)$
A ends before the end of B		$ends(A) \leq ends(B)$
A ends after the end of B		$end(A) \geq end(B)$
Delay synchronism	$A \left\{ \begin{array}{l} starts \\ ends \end{array} \right. \left\{ \begin{array}{l} exactly \\ at\ least \\ at\ most \end{array} \right. X\ years \left\{ \begin{array}{l} before \\ after \end{array} \right. \left\{ \begin{array}{l} start\ of \\ end\ of \end{array} \right. B$	

Figure 18 Additional CHRONOLOG synchronisms. In the images, time flows from above to below.

Appendix B. Detailed CHRONOLOG to OxCal Conversion Rules

Table 4   Detailed CHRONOLOG to OxCal conversion rules (sequences and bounds).

SEQUENCES AND BOUNDS

CHRONOLOG	Conversion to OxCal	Comments

Sequence S, containing two periods: A and B	Sequence("S")
	{
	    After("", −50000);
	    Boundary("A_start");
	    Phase("A")
	    {
	    };
	    Boundary("B_start");
	    Phase("B")
	    {
	    };
	    Boundary("B_end");
	    Before("", 1950);
	};
		• One phase per period
		• One boundary between each phase
		• If no earliest start nor ^{14}C date is provided for the first period, an earliest date of 50,000 BCE is set for the sequence.
		• If no latest end nor ^{14}C date is provided for the last period, a latest date of 1950 CE is set for the sequence.

Boundary ≥ X	After("", X.0); Boundary();	X.0 rather than X, to signify the start of the year.
Boundary ≤ X	Boundary(); Before("", X.999);	X.999 rather than X, to signify the end of the year.
Boundary = X	Boundary("", CE(X));	Using CE also for negative dates (year 0 included).

Duration ≥ X	Sequence("S")
	{
	  Phase("A")
	  {
	   Interval("duration");
	  };
	};
	duration > X;	

Duration ≤ X	Sequence("S")
	{
	  Phase("A")
	  {
	   Interval("duration");
	  };
	};
	duration < X;	

Duration = X	Phase()
	{
	  Sequence(S)
	  {
	  Boundary("A_start");
	  Phase("A")
	  {
	  };
	  Boundary("=A_end");
	  };
	};
	A_end = A_start + X;	

Table 5 Detailed CHRONOLOG to OxCal conversion rules (synchronisms).

SYNCHRONISMS

Note: all the CHRONOLOG synchronisms involve a combination of the basic cases shown below

CHRONOLOG	Conversion to OxCal	Comments
Equality between boundaries: $start(A) = start(B)$	A_start = B_start + 0	
Inequality between boundaries (lower than): $start(A) \leq start(B)$	A_start < B_start + 1	The "+1" is used to simulate CHRONOLOG's non-strict inequality ($\leq$) using OxCal's strict inequality ($<$).
Inequality between boundaries (greater than): $start(A) \geq start(B)$	A_start > B_start - 1	The "-1" is used to simulate CHRONOLOG's non-strict inequality ($\geq$) using OxCal's strict inequality ($>$).
"Delay synchronism": A starts at most 5 years after start of B	A_start < B_start + 6	Other types of synchronisms are implemented in the same way. Note the use of "< 6" to signify "$\leq$ 5".

Table 6 Detailed CHRONOLOG to OxCal conversion rules (radiocarbon dates).

RADIOCARBON DATES

CHRONOLOG	Conversion to OxCal	Comments

^{14}C date within a period	Phase("A")
	{
	  R_Date(800,25);
	};	

^{14}C dates within a boundary	Boundary("A_start")
	{
	  R_Date(800,25);
	};
		For an example of ^{14}C dates within a boundary, see the script in Garfinkel et al. 2019 (supplementary material).

Outlier analysis	Phase("A")
	{
	  R_Date(777,30)
	  {
	    Outlier(0.05);
	  };
	};
		Uses the following outlier model: Outlier_Model("General",T(5), U(0,4),"t"); (see Bronk Ramsey 2009b: 1028).

Appendix C. Details of the case studies

Case-study 1: Egyptian 18th Dynasty

Radiocarbon Dates

Table 7　Radiocarbon determinations included in the first case-study (Egyptian 18th dynasty). The radiocarbon determinations for the kings are from Bronk Ramsey et al. 2010 (excluding outlier samples no. 18520, 19550, 19004, 19263, VERA-4686, VERA-4686B, 20482, and 18954). Those of the tomb of Sennefer are from Quiles et al. 2013 ("Bouquet 1" only). Dates with an asterisk are combined.

Reign/tomb	Lab no.	^{14}C date
Amenhotep II	18507	3165 ± 32
Thutmoses IV	19549	3084 ± 26
Amenhotep III	18510	3040 ± 30
	19548	3118 ± 27
Amenhotep IV	18057	3082 ± 29
	18407	3096 ± 28
	18512	3051 ± 27*
	18412	3064 ± 28*
	18953	3092 ± 27
	18955	3115 ± 30*
	VERA-4687	3094 ± 37*
	VERA-4687B	3070 ± 37*
	18956	3028 ± 27
	VERA-4685	3096 ± 34*
	VERA-4685B	3116 ± 35*
Tutankhamun	17868	3065 ± 31
	18950	3138 ± 28
	18951	3137 ± 29
	18952	3117 ± 29
	19003	3106 ± 26
	19132	3133 ± 29
Sennefer tomb ("Bouquet 1")	SacA 15966	3101 ± 34*
	SacA 18758	3128 ± 26*
	SacA 15967	3123 ± 39*
	SacA 18759	3089 ± 26*
	SacA 15968	3047 ± 36*
	SacA 18760	3042 ± 29*

OxCal Script

```
Options() {
 Resolution = 1;
};
Plot() {
 Outlier_Model("General",  T(5),  U(0,  4),
"t");
 Phase() {
  Sequence("Dynasty_18") {
   Boundary("Amenhotep_II_start");
   Phase("Amenhotep_II") {
    Interval("Amenhotep_II_int");
    R_Date("18507", 3165, 32) {
     Outlier(0.05);
    };
   };
   Boundary("Thutmoses_IV_start");
   Phase("Thutmoses_IV") {
    Interval("Thutmoses_IV_int");
    R_Date("19549", 3084, 26) {
     Outlier(0.05);
    };
   };
   Boundary("Amenhotep_III_start");
   Phase("Amenhotep_III") {
    Interval("Amenhotep_III_int");
    R_Date("18510", 3040, 30) {
     Outlier(0.05);
    };
    R_Date("19548", 3118, 27) {
     Outlier(0.05);
    };
   };
   Boundary("Amenhotep_IV_start");
   Phase("Amenhotep_IV") {
    Interval("Amenhotep_IV_int");
    R_Date("18057", 3082, 29) {
     Outlier(0.05);
    };
    R_Date("18407", 3096, 28) {
     Outlier(0.05);
    };
    R_Combine("Ahp_IV_Combine_1") {
     Outlier(0.05);
     R_Date("18512", 3051, 27);
     R_Date("18412", 3064, 28);
    };
    R_Date("18953", 3092, 27) {
     Outlier(0.05);
    };
    R_Combine("Ahp_IV_Combine_4") {
     Outlier(0.05);
     R_Date("18955", 3115, 30);
     R_Date("VERA-4687", 3094, 37);
     R_Date("VERA-4687B", 3070, 37);
    };
    R_Date("18956", 3028, 27) {
     Outlier(0.05);
    };
    R_Combine("Ahp_IV_Combine_5") {
     Outlier(0.05);
     R_Date("VERA-4685", 3096, 34);
     R_Date("VERA-4685B", 3116, 35);
    };
   };
   Boundary("Neferneferuaten_start");
   Phase("Neferneferuaten") {
    Interval("Neferneferuaten_int");
   };
   Boundary("Tutankhamun_start");
   Phase("Tutankhamun") {
    Interval("Tutankhamun_int");
    R_Date("17868", 3065, 31) {
     Outlier(0.05);
    };
    R_Date("18950", 3138, 28) {
     Outlier(0.05);
    };
    R_Date("18951", 3137, 29) {
     Outlier(0.05);
    };
    R_Date("18952", 3117, 29) {
     Outlier(0.05);
    };
    R_Date("19003", 3106, 26) {
     Outlier(0.05);
    };
    R_Date("19132", 3133, 29) {
     Outlier(0.05);
      };
    };
    Boundary("Ay_start");
    Phase("Ay") {
     Interval("Ay_int");
    };
    Boundary("Horemheb_start");
    Phase("Horemheb") {
     Interval("Horemheb_int");
    };
    Boundary("Horemheb_end");
    Before("", -1200.999);
   };
   Amenhotep_II_int > 22;
   Amenhotep_II_int < 25;
   Thutmoses_IV_int > 9;
   Thutmoses_IV_int < 10;
   Amenhotep_III_int > 37;
   Amenhotep_III_int < 38;
   Amenhotep_IV_int > 16;
   Amenhotep_IV_int < 18;
   Neferneferuaten_int > 3;
```

```
Neferneferuaten_int < 5;
Tutankhamun_int > 9;
Tutankhamun_int < 10;
Ay_int > 3;
Ay_int < 4;
Horemheb_int > 13;
Horemheb_int < 27;
Sequence("Sennefer_Tomb") {
 Boundary("Sennefer_Tomb_start");
 Phase("Sennefer_Tomb") {
  R_Combine("Bouquet1") {
   Outlier(0.05);
   R_Date(3101, 34);
       R_Date(3089, 26);
       R_Date(3123, 39);
       R_Date(3128, 26);
       R_Date(3047, 36);
       R_Date(3042, 29);
      };
     };
   Boundary("Sennefer_Tomb_end");
  };
 };
 Sennefer_Tomb_start > Tutankhamun_start -1;
 Sennefer_Tomb_end < Horemheb_start + 1;
};
```

Case-study 2: Late Helladic to Proto-Geometric Aegean Chronology

Radiocarbon Dates

Table 8   Radiocarbon dates for the Aegean chronology from the late Late Helladic IIIB to the Early Proto-Geometric (adapted from Fantalkin et al. 2015: 28, their Table 1). Dates with an asterisk are combined.

Site/stratum	Lab no.	^{14}C date
Megiddo K8	5501	2920 ± 40
	5502	2945 ± 40
	5503	2980 ± 40
	5882	2990 ± 55
Megiddo K7	5504	2935 ± 40
	5884	2980 ± 50
Beth Shean VII	2594	2925 ± 25
	2597	2985 ± 25
	2156	2910 ± 50
Maroni	4647	2969 ± 40
	8265	2960 ± 35
	8266	2985 ± 35
	8267	2940 ± 35
	8324	2930 ± 40
Tell Tweini	281582	3000 ± 40
	281576	2990 ± 40
	281572	2950 ± 40
	281590	2950 ± 40
	281591	2950 ± 40
	281570	2910 ± 40
Apliki	33450	2990 ± 45
	33451	2960 ± 45
	33452	2930 ± 60
	33452a	2945 ± 50
	33453	2960 ± 50
	33454	2955 ± 65
Beth Shean Lower VI	2323	2900 ± 45
	2325	2980 ± 40
	2527	2920 ± 25
	2596	2960 ± 25
Tel Miqne VIIB	4286.3	2950 ± 55*
	4286.4	2900 ± 40*
	4286.5	2870 ± 60*
Tel Miqne VIB	4283.3	2915 ± 45*
	4283.4	2960 ± 45*
	4283.5	2880 ± 45*
	4282a, aa	2883 ± 26
	4282.3–5	2872 ± 27
Tel Miqne VB	4284.3	2835 ± 45
	4284.4	2830 ± 45
Beth Shemesh 6	3934.3	2830 ± 50*
	3934.4	2925 ± 50*
	3934.5	2810 ± 50*
Beth Shemesh 5	3936.3-5	2835 ± 32
Megiddo K-4, H-9	3945a,aa	2880 ± 30
	3946a,aa	2910 ± 25
	RTK-6273	2881 ± 26
	18163a	2864 ± 40

OxCal Script

```
Plot() {
 Phase() {
  Sequence("Archaeological_periods") {
   After("",-50000);
   Boundary("Late_LH_IIIB_start");
   Phase("Late_LH_IIIB") {};
   Boundary("LH_IIIC_E1_start");
   Phase("LH_IIIC_E1") {};
   Boundary("LH_IIIC_E2M1_start");
   Phase("LH_IIIC_E2M1") {};
   Boundary("LH_IIIC_M2_start");
   Phase("LH_IIIC_M2") {};
   Boundary("LH_IIIC_Late_start");
   Phase("LH_IIIC_Late") {};
   Boundary("EPG_start");
   Phase("EPG") {};
   Boundary("EPG_end");
   Before("", 1950);
  };
  Sequence("Megiddo_K8K7") {
   Boundary("Megiddo_K8_start");
   Phase("Megiddo_K8") {
    R_Date(2920, 40);
    R_Date(2945, 40);
    R_Date(2980, 40);
    R_Date(2990, 55);
   };
   Boundary("Megiddo_K7_start");
   Phase("Megiddo_K7") {
    R_Date(2935, 40);
    R_Date(2980, 50);
   };
   Boundary("Megiddo_K7_end");
  };
  Sequence("Beth_Shean_VII") {
   Boundary("BS_VII_start");
   Phase("BS_VII") {
    R_Date(2985, 25);
    R_Date(2925, 25);
    R_Date(2910, 50);
   };
   Boundary("BS_VII_end");
  };
  Sequence("Maroni") {
   Boundary("Maroni_start");
   Phase("Maroni") {
    R_Date(2969, 40);
    R_Date(2960, 35);
    R_Date(2985, 35);
    R_Date(2940, 35);
    R_Date(2930, 40);
   };
   Boundary("Maroni_end");
  };
   Sequence("Tell_Tweini") {
    Boundary("Tell_Tweini_start");
    Phase("Tell_Tweini") {
     R_Date(3000, 40);
     R_Date(2990, 40);
     R_Date(2950, 40);
     R_Date(2950, 40);
     R_Date(2950, 40);
     R_Date(2910, 40);
    };
    Boundary("Tell_Tweini_end");
    Before("", -1175.999);
   };
  Sequence("Apliki") {
   Boundary("Apliki_start");
   Phase("Apliki") {
    R_Date(2990, 45);
    R_Date(2960, 45);
    R_Date(2930, 60);
    R_Date(2945, 50);
    R_Date(2960, 50);
    R_Date(2955, 65);
   };
   Boundary("Apliki_end");
  };
  Sequence("Beth_Shean_Lower_VI") {
   Boundary("BS_Lower_VI_start");
   Phase("BS_Lower_VI") {
    R_Date(2900, 45);
    R_Date(2980, 40);
    R_Date(2920, 25);
    R_Date(2960, 25);
   };
   After("",-1150.0);
   Boundary("BS_Lower_VI_end");
  };
  Sequence("Miqne_VIIB") {
   Boundary("Miqne_VIIB_start");
   Phase("Miqne_VIIB") {
    R_Combine("4286.3--5") {
     R_Date(2950, 55);
     R_Date(2900, 40);
     R_Date(2870, 60);
    };

   };
   Boundary("Miqne_VIIB_end");
  };
  Sequence("Beth_Shemesh_6_5") {
   Boundary("Shemesh_6_start");
   Phase("Shemesh_6") {
    R_Combine("3934.3--5") {
     R_Date(2830, 50);
     R_Date(2925, 50);
     R_Date(2810, 50);
    };
```

```
};                                                  R_Date(2881, 26);
Boundary("Shemesh_5_start");                        R_Date(2864, 40);
Phase("Shemesh_5") {                                };
 R_Date(2835, 32);                                    Boundary("Megiddo_K4H9_end");
};                                                   };
Boundary("Shemesh_5_end");                          };
};                                                  Megiddo_K8_start > Late_LH_IIIB_start -1;
Sequence("Miqne_VIB_VB") {                          Megiddo_K7_start < LH_IIIC_E1_start + 1;
Boundary("Miqne_VIB_start");                        Megiddo_K7_start > Late_LH_IIIB_start -1;
Phase("Miqne_VIB") {                                Megiddo_K7_end < LH_IIIC_E1_start + 1;
 R_Combine("4283.3--5") {                           Maroni_start > Late_LH_IIIB_start - 1;
  R_Date(2915, 45);                                 Maroni_end < LH_IIIC_E1_start + 1;
  R_Date(2960, 45);                                 Tell_Tweini_start > LH_IIIC_E1_start - 1;
  R_Date(2880, 45);                                 Tell_Tweini_end < LH_IIIC_E2M1_start + 1;
 };                                                 Apliki_start > LH_IIIC_E1_start - 1;
                                                    Apliki_end < LH_IIIC_E2M1_start + 1;
 R_Date(2883, 26);                                  Miqne_VIIB_start > LH_IIIC_M2_start - 1;
 R_Date(2872, 27);                                  Miqne_VIIB_end < LH_IIIC_Late_start + 1;
};                                                  Miqne_VIB_start > LH_IIIC_Late_start - 1;
Boundary("Miqne_VIA_start");                        Miqne_VIA_start < EPG_start + 1;
Phase("Miqne_VIA") {};                              BS_VII_start > Late_LH_IIIB_start - 1;
Boundary("Miqne_VB_start");                         BS_VII_end < LH_IIIC_E1_start + 1;
Phase("Miqne_VB") {                                 BS_Lower_VI_start > LH_IIIC_E2M1_start -1;
 R_Date(2835, 45);                                  BS_Lower_VI_end < LH_IIIC_M2_start + 1;
 R_Date(2830, 45);                                  Shemesh_6_start > LH_IIIC_Late_start - 1;
};                                                  Shemesh_5_start < EPG_start + 1;
Boundary("Miqne_VB_end");                           Shemesh_5_start > LH_IIIC_Late_start - 1;
};                                                  Shemesh_5_end < EPG_start + 1;
Sequence("Megiddo_K4_H9") {                         Megiddo_K4H9_start > EPG_start - 1;
 Boundary("Megiddo_K4H9_start");                    Megiddo_K4H9_end < EPG_end + 1;
 Phase("Megiddo_K4H9") {                            Miqne_VB_start > LH_IIIC_Late_start - 1;
  R_Date(2880, 30);                                 Miqne_VB_end < EPG_start + 1;
  R_Date(2910, 25);                                 };
```

Case-study 3: Iron I/Iron II Transition in the Southern Levant

Radiocarbon Dates

Table 9   Radiocarbon dates for the Iron I/II transition in the southern levant (adapted from Mazar and Bronk Ramsey 2008, model C3, excluding additional samples HM3 and QS6). In cases of discrepancies (samples MG5, MG9, Y1, Y3, GrN-26121, GrN-18825) between Mazar and Bronk Ramsey's OxCal script (p. A23) and their table (Table 2, p. 164), we have followed the script. Regarding the Tel Miqne samples (originating in Stratum V but initially reported as coming from Stratum IV), see Mazar and Bronk Ramsey 2008, note 3. Dates with an asterisk are combined.

Site/stratum	Code (Mazar and Bronk Ramsey 2008)	^{14}C date
Megiddo K-6	MG1	2893 ± 27*
	MG2	2894 ± 23*
	MG3	2968 ± 30*
	MG4	2918 ± 22*
Hazor XII/XI	HZ3	3060 ± 50*
	HZ8	3060 ± 30*
Tel Miqne V	MQ1	2883 ± 26*
	MQ2	2872 ± 27*
Tel Rehov D-4	R1	2845 ± 25*
	R2	2913 ± 45*
	GrN-26121	2890 ± 30*
	GrN-18825	2870 ± 50*
Shiloh V	SH1	2868 ± 20
	SH2	2854 ± 25
	SH3	2897 ± 23
	SH4	2959 ± 28
Megiddo VIA (=K4) destruction	MG5	2925 ± 25*
	MG6	2864 ± 40*
	MG9	2760 ± 26
	MG10	2765 ± 25
	MG11	2845 ± 25
	MG12	2855 ± 25
Megiddo VIA (=K4) pre-destruction	MG7	2880 ± 30
	MG8	2910 ± 25
Yoqneam XVII	Y1	2866 ± 15*
	Y2	2866 ± 33*
	Y3	2776 ± 15*
	Y4	2817 ± 26*
	Y5	2818 ± 29*
	Y11	2925 ± 38*
	Y12	2897 ± 38*
	Y13	2929 ± 54*
	Y6	2926 ± 30*
	Y7	2824 ± 30*
Tell Keisan 9a-b	K8	2855 ± 29
Tell Qasile X	QS3	2911 ± 26*
	QS4	2853 ± 25*
	QS5	2895 ± 25
	QS7	2800 ± 25
	QS8	2882 ± 28
	QS9	2864 ± 40*
	QS10	2818 ± 38*
	QS11	2897 ± 44*

Table 9 (*Continued*)

Site/stratum	Code (Mazar and Bronk Ramsey 2008)	[14]C date
Tel Dor D2/9–10 IrI(l)	D1	2803 ± 16
	D2	2783 ± 22
Tel Hadar IV	HD1	2791 ± 52
	HD2	2853 ± 13
Tel Rehov D-3 Locus 2862	R3a	2800 ± 20*
	R3b	2835 ± 45*
	GrN-26119	2720 ± 30*
Tel Rehov D-2	R4	2757 ± 20*
	R5	2770 ± 23*
	R5a	2805 ± 15*
Tel Rehov E-1b/E-1823	R6	2685 ± 25*
	R7	2678 ± 20*
	R8	2754 ± 24
Aphek X-8	A1	2667 ± 20
Tel Dor D2/8c	D3	2757 ± 18
	D4	2764 ± 22
	D5	2779 ± 24
Tel Dor D2/8b	D6	2750 ± 23
Megiddo H-5	MG13	2783 ± 32*
	MG14	2859 ± 34*
	MG15	2695 ± 50
Horbat Rosh Zayit IIa	RZ1	2745 ± 30*
	RZ2	2755 ± 22*
	RZ6	2689 ± 28*
	RZ3	2745 ± 30*
	RZ4	2729 ± 37*
	RZ5	2692 ± 31*
	RZ7	2709 ± 15
Bethsaida VI	BD1	2820 ± 35*
	BD2	2786 ± 25*
Hazor IXA	HZ15	2697 ± 24*
	HZ16	2689 ± 27*
Tell el-Hammah "mid" level	HM4	2675 ± 23
	HM5	2688 ± 25
	HM6	2687 ± 20
	HM7	2701 ± 22
	HM8	2609 ± 21
	HM9	2587 ± 23
	HM10	2634 ± 23
	HM11	2636 ± 23
	HM12	2722 ± 24
	HM13	2728 ± 28
Tell es-Safi temporary 4	SF1	2661 ± 30
	SF2	2723 ± 18

OxCal Script

```
Options() {
 Resolution = 1;
};
Plot() {
 Phase() {
  Sequence("Main") {
   After("", -50000);
   Boundary("Iron_I_start");
   Phase("Iron_I") {};
   Boundary("Iron_II_start");
   Phase("Iron_II") {};
   Boundary("Iron_II_end");
   Before("", 1950);
  };
  Sequence("Late_I") {
   After("", -50000);
   Boundary("Late_I1_start");
   Phase("Late_I1") {};
   Boundary("=Late_I1_end");
   Before("", 1950);
  };
  Sequence("Meg_K6") {
   Boundary("Meg_K6_start");
   Phase("Meg_K6") {
    R_Combine() {
     R_Date(2893, 27);
     R_Date(2894, 23);
    };
    R_Combine() {
     R_Date(2968, 30);
     R_Date(2918, 22);
    };
   };
   Boundary("Meg_K6_end");
  };
  Sequence("Hazor_XII_XI") {
   Boundary("Hazor_XII_XI_start");
   Phase("Hazor_XII_XI") {
    R_Combine("HZ3_&_8") {
     R_Date(3060, 50);
     R_Date(3060, 30);
    };
   };
   Boundary("Hazor_XII_XI_end");
  };
  Sequence("Miqne_IV") {
   Boundary("Miqne_IV_start");
   Phase("Miqne_IV") {
    R_Combine() {
     R_Date(2883, 26);
     R_Date(2872, 27);
    };
   };
   Boundary("Miqne_IV_end");
  };
  Sequence("Rehov_D4") {
   Boundary("Rehov_D4_start");
   Phase("Rehov_D4") {
    R_Combine() {
     R_Date(2845, 25);
     R_Date(2913, 45);
     R_Date(2890, 30);
     R_Date(2870, 50);
    };
   };
   Boundary("Rehov_D4_end");
  };
  Sequence("Shiloh_V") {
   Boundary("Shiloh_V_start");
   Phase("Shiloh_V") {
    R_Date(2868, 20);
    R_Date(2854, 25);
    R_Date(2897, 23);
    R_Date(2959, 28);
   };
   Boundary("Shiloh_V_end");
  };
  Sequence("Rehov_D2") {
   Boundary("Rehov_D2_start");
   Phase("Rehov_D2") {
    R_Combine() {
     R_Date(2757, 20);
     R_Date(2770, 23);
     R_Date(2805, 15);
    };
   };
   Boundary("Rehov_D2_end");
  };
  Sequence("Rehov_E1b") {
   Boundary("Rehov_E1b_start");
   Phase("Rehov_E1b") {
    R_Combine() {
    R_Date(2685, 25);
    R_Date(2678, 20);
    };
    R_Date(2754, 24);
   };
   Boundary("Rehov_E1b_end");
  };
  Sequence("Aphek_X8") {
   Boundary("Aphek_X8_start");
   Phase("Aphek_X8") {
    R_Date(2667, 20);
   };
   Boundary("Aphek_X8_end");
  };
  Sequence("Dor_D28bc") {
   Boundary("Dor_D28bc_start");
   Phase("Dor_D28bc") {
    R_Date(2757, 18);
```

```
   R_Date(2764, 22);
   R_Date(2779, 24);
   R_Date(2750, 23);
  };
 Boundary("Dor_D28bc_end");
};
Sequence("Meg_H5") {
 Boundary("Meg_H5_start");
 Phase("Meg_H5") {
  R_Combine() {
   R_Date(2783, 32);
   R_Date(2859, 34);
  };
  R_Date(2695, 50);
 };
 Boundary("Meg_H5_end");
};
Sequence("Zayit_IIa") {
 Boundary("Zayit_IIa_start");
 Phase("Zayit_IIa") {
  R_Combine() {
   R_Date(2745, 30);
   R_Date(2755, 22);
   R_Date(2689, 28);
  };
  R_Combine() {
   R_Date(2745, 30);
   R_Date(2729, 37);
   R_Date(2692, 31);
  };
  R_Date(2709, 15);
 };
 Boundary("Zayit_IIa_end");
};
 Sequence("Bethsaida_VI") {
  Boundary("Bethsaida_VI_start");
  Phase("Bethsaida_VI") {
   R_Combine() {
   R_Date(2820, 35);
   R_Date(2786, 25);
  };
 };
 Boundary("Bethsaida_VI_end");
};
Sequence("Hazor_IXA") {
 Boundary("Hazor_IXA_start");
 Phase("Hazor_IXA") {
  R_Combine() {
   R_Date(2697, 24);
   R_Date(2689, 27);
  };
 };
 Boundary("Hazor_IXA_end");
};

Sequence("el_Hammah") {
 Boundary("el_Hammah_start");
 Phase("el_Hammah") {
  R_Date(2675, 23);
  R_Date(2688, 25);
  R_Date(2687, 20);
  R_Date(2701, 22);
  R_Date(2609, 21);
  R_Date(2587, 23);
  R_Date(2634, 23);
  R_Date(2636, 23);
  R_Date(2722, 24);
  R_Date(2728, 28);
 };
 Boundary("el_Hammah_end");
};
Sequence("Safi_IV") {
 Boundary("Safi_IV_start");
 Phase("Safi_IV") {
  R_Date(2661, 30);
  R_Date(2723, 18);
 };
 Boundary("Safi_IV_end");
};
Sequence("Meg_VIA_pre") {
 Boundary("Meg_VIA_pre_start");
 Phase("Meg_VIA_pre") {
  R_Date(2880, 30);
  R_Date(2910, 25);
 };
 Boundary("Meg_VIA_pre_end");
};
Sequence("Meg_VIA_destr") {
 Boundary("Meg_VIA_destr_start");
 Phase("Meg_VIA_destr") {
  R_Combine() {
   R_Date(2925, 25);
   R_Date(2864, 40);
  };
  R_Date(2760, 26);
  R_Date(2765, 25);
  R_Date(2845, 25);
  R_Date(2855, 25);
 };
 Boundary("Meg_VIA_destr_end");
};
Sequence("Yoqneam_XVII") {
 Boundary("Yoqneam_XVII_start");
 Phase("Yoqneam_XVII") {
  R_Combine() {
   R_Date(2866, 15);
   R_Date(2866, 33);
  };
```

```
        R_Combine() {                          R_Date(2835, 45);
         R_Date(2776, 15);                      R_Date(2720, 30);
         R_Date(2817, 26);                     };
         R_Date(2818, 29);                    };
        };                                    Boundary("Rehov_D3_end");
        R_Combine() {                        };
         R_Date(2926, 30);                  };
         R_Date(2824, 30);                  Late_I1_end = Iron_II_start + 0;
        };                                  Meg_K6_start > Iron_I_start - 1;
        R_Combine() {                       Meg_K6_end < Iron_II_start + 1;
         R_Date(2925, 38);                  Hazor_XII_XI_start > Iron_I_start - 1;
         R_Date(2897, 38);                  Hazor_XII_XI_end < Iron_II_start + 1;
         R_Date(2929, 54);                  Miqne_IV_start > Iron_I_start - 1;
        };                                  Miqne_IV_end < Iron_II_start + 1;
       };                                   Rehov_D4_start > Iron_I_start - 1;
       Boundary("Yoqneam_XVII_end");        Rehov_D4_end < Iron_II_start + 1;
      };                                    Shiloh_V_start > Iron_I_start - 1;
     Sequence("Qasile_X") {                 Shiloh_V_end < Iron_II_start + 1;
      Boundary("Qasile_X_start");           Rehov_D2_start > Iron_II_start - 1;
      Phase("Qasile_X") {                   Rehov_D2_end < Iron_II_end + 1;
       R_Combine() {                        Aphek_X8_start > Iron_II_start - 1;
        R_Date(2911, 26);                   Aphek_X8_end < Iron_II_end + 1;
        R_Date(2853, 25);                   Rehov_E1b_start > Iron_II_start - 1;
        R_Date(2864, 40);                   Rehov_E1b_end < Iron_II_end + 1;
        R_Date(2818, 38);                   Dor_D28bc_start > Iron_II_start - 1;
        R_Date(2897, 44);                   Dor_D28bc_end < Iron_II_end + 1;
       };                                   Meg_H5_start > Iron_II_start - 1;
       R_Date(2895, 25);                    Meg_H5_end < Iron_II_end + 1;
       R_Date(2800, 25);                    Zayit_IIa_start > Iron_II_start - 1;
       R_Date(2882, 28);                    Zayit_IIa_end < Iron_II_end + 1;
      };                                    Bethsaida_VI_start > Iron_II_start - 1;
      Boundary("Qasile_X_end");            Bethsaida_VI_end < Iron_II_end + 1;
     };                                     Hazor_IXA_start > Iron_II_start - 1;
     Sequence("Dor") {                      Hazor_IXA_end < Iron_II_end + 1;
      Boundary("Dor_start");                el_Hammah_start > Iron_II_start - 1;
      Phase("Dor") {                        el_Hammah_end < Iron_II_end + 1;
       R_Date(2803, 16);                    Safi_IV_start > Iron_II_start - 1;
       R_Date(2783, 22);                    Safi_IV_end < Iron_II_end + 1;
      };                                    Meg_VIA_pre_start > Late_I1_start - 1;
      Boundary("Dor_end");                  Meg_VIA_pre_end < Late_I1_end + 1;
     };                                     Meg_VIA_destr_start > Late_I1_start - 1;
     Sequence("Hadar_IV") {                 Meg_VIA_destr_end < Late_I1_end + 1;
      Boundary("Hadar_IV_start");           Hadar_IV_start > Late_I1_start - 1;
      Phase("Hadar_IV") {                   Hadar_IV_end < Late_I1_end + 1;
       R_Date(2791, 52);                    Yoqneam_XVII_start > Late_I1_start - 1;
       R_Date(2853, 13);                    Yoqneam_XVII_end < Late_I1_end + 1;
      };                                    Qasile_X_start > Late_I1_start - 1;
      Boundary("Hadar_IV_end");             Qasile_X_end < Late_I1_end + 1;
     };                                     Dor_start > Late_I1_start - 1;
     Sequence("Rehov_D3") {                 Dor_end < Late_I1_end + 1;
      Boundary("Rehov_D3_start");           Rehov_D3_start > Late_I1_start - 1;
      Phase("Rehov_D3") {                   Rehov_D3_end < Late_I1_end + 1;
       R_Combine() {                       };
        R_Date(2800, 20);
```