
3.1 Introduction

This chapter will focus on the Minerva 2 model, a global matching model of recognition memory. Recall from Chap. 1 that the recognition memory task takes place in at least two phases. In the first study phase, people are given a list of items to study (e.g., words) and instructed to commit them to memory. Following the study phase, the subject might perform some filler task, such as completing a puzzle. Often, these filler tasks are used to either remove recency effects or to equate the retention interval across different conditions. In the second test phase, the subject is presented with a probe item and asked to respond either “old,” meaning that the subject believes the probe was on the previously studied list, or “new,” meaning that the subject believes the probe was not on the previously studied list. The proportion of old responses to targets (hit rates) can be plotted as a function of the proportion of old responses to distractors (false alarm rates), producing the receiver operating characteristic (ROC) curve [71, 72].

After a brief description of the model, the tutorial will be broken into two parts. In the first part, we will fit the Minerva 2 model to simulated data using three likelihood-free methods: kernel-based Markov chain Monte Carlo, probability density approximation (PDA), and analytic expressions derived by Sheu [37]. After fitting the model using each technique, we will compare the three methods by examining the estimated marginal posterior distributions for each of the model parameters and the computational time for each method. The simulation study will also allow us to better understand the model by permitting us to examine the joint posteriors for each parameter generated using the PDA method.

In the second part of the tutorial, we will fit a hierarchical version of the Minerva 2 model to data from [4] by blending two likelihood-free techniques (PDA and Gibbs ABC). Prior to fitting the model, we will describe the hierarchical framework in detail, and after determining if the fit is adequate, we will examine the posterior distributions of specific parameters to gain a better understanding of both the effectiveness of the experimental manipulation and the predictions made by the model.

3.2 MINERVA 2 Model

Minerva 2 is a member of the class of *global matching models*, which includes memory models such as the Theory of Distributed Associative Memory model (TODAM) [73], the Search of Associative Memory model (SAM) [74], and the Matrix model [75, 76]. Global matching refers to a retrieval process in which a probe item is compared against the contents of memory, producing a single summed *familiarity* value that indexes the similarity of the probe to the contents of memory. The familiarity value is subsequently compared against a decision criterion to produce an “old” or “new” decision [77]. While all of the global matching models make mathematically identical predictions under some circumstances [78], Minerva 2 possesses some unique properties that differentiate it from the other models, such as a non-linear activation function that ensures that traces more similar to the probe have a greater contribution to the familiarity calculation.

We will describe the mathematics of Minerva 2 briefly. Readers interested in a more detailed description of the model and its predictions should consult the original publications [36, 79]. In Minerva 2, items are represented as a vector of η features that take the values of 1, 0, or -1 with equal probability. When an item is presented for study, a new trace vector is created in memory that contains features from the original item vector that are copied with probability L . The probability L is called the learning rate of the model. If a feature from the item vector is not copied into memory, the item trace has a 0 stored for that feature. After a set of items have been studied, the contents of memory are represented by a matrix M , which contains all of the trace vectors created in the study episode. Over time, correctly copied features may revert to 0 with probability δ . The probability δ is called the decay rate, and it is this decay rate that models the effects of a retention interval, with higher decay rates used for longer study-test intervals [36, 79].

Global matching operates in Minerva 2 by comparing the probe vector P against each of the trace vectors in the memory matrix M . The similarity S_i between the probe and trace vector i is calculated as

$$S_i = \sum_{j=1}^N \frac{P_j T_{ij}}{\epsilon_i}, \quad (3.1)$$

where T_i is the i th trace vector in M , j is the j th feature in the comparison between the probe and trace vectors, and ϵ_i is the number of features where $P_j \neq 0$ and

$T_{i,j} \neq 0$. Values of S_i are equal to 1 if the probe vector is identical to the trace vector and 0 if the two vectors are orthogonal to each other. The extent to which a probe item activates the traces in memory is computed as the activation

$$A_i = S_i^3. \quad (3.2)$$

The cubing of the similarity values produces a non-linear relationship between similarity and activation: activation values are highest for traces that are most similar to the probe vector [79]. The activation values are subsequently summed to produce a value of “echo intensity” I (i.e., familiarity), so

$$I = \sum_{i=1}^M A_i. \quad (3.3)$$

The echo intensity I is compared to a decision criterion C . If I is greater than C , familiarity is high enough to produce an “old” response. If not, a “new” response is made. Because echo intensity is greatest when the similarity between the probe and the traces is highest, the values of I tend to be higher for targets: There is a higher expected similarity between the target trace vector and its own probe item than between a trace vector and an unrelated probe item. The distributions of intensity values tend to have higher variance for targets than for distractors, a difference that arises from the non-linear activation function and the probabilistic encoding of features [80]. Performance tends to be worse for longer study lists than for shorter ones because, as the number of trace vectors in M is increased, the variance of the echo intensity values increases for both targets and distractors, resulting in decreased discriminability. This list-length effect is predicted by other global matching models for similar reasons [77].

Minerva 2 and the other global matching models were challenged by a series of findings in the recognition memory literature. One of which was the *null list strength effect* [81], in which strengthening a subset of studied items by increasing study time or number of presentations does not decrease performance for the other non-strengthened items. Another was the *mirror effect* [82], whereby some manipulations produce opposite effects on the hit and false alarm rates. The global matching models were not able to capture these effects without modification, and so a newer generation of Bayesian recognition memory models were developed. These models include the Retrieving Effectively from Memory model (REM; [83]), the Subjective Likelihood in Memory model [84], and the Bind Cue Decide Model of Episodic Memory (BCDMEM; [85]), all of which can be described as the dominant theoretical models of recognition memory currently. BCDMEM and REM will become important in a later application.

Minerva 2 has been used to explain many memory-related phenomena. Arndt and Hirshman [86] found that Minerva 2 was able to successfully predict a number of relations between true and false recognition. They further demonstrated that the non-linear activation function was specifically responsible for the success of

these predictions, and that other models such as TODAM or the Matrix model cannot do the same. A dual-process variant of Minerva 2 was used by Benjamin [87] to explain dissociations between item and source recognition in the aging literature. Minerva 2 was used to develop successful models of judgment and decision making, including judgments of likelihood (MINERVA DM) [88] and hypothesis generation (HyGene) [89]. It has also been applied to semantic memory phenomena, such as performance on lexical decision tasks [90] and the formation of lexical representations [91].

3.2.1 Implementing the Model

The Minerva 2 model is a simulation model, meaning that exact equations have not yet been produced for evaluating the likelihood function. As such, to fit the model to data in this chapter, we rely on likelihood-free techniques. The minimum requirement of these techniques is that we be able to simulate data from the model, which means we must first prepare computer code to implement the equations in the preceding section.

First, to offload some complexity of the model simulation code, we can write a separate function to generate the features that represent the items used at study and test.

```
1 init=function(N,p) sample(c(-1,0,1),N,replace=T,prob=p)
```

The `init` function simply samples the features $\{-1, 0, 1\}$ with probability determined by the variable `p`. The end result is a vector of randomly selected features of length `N`, corresponding to η . With a simple function for generating feature values, the next block of code creates a function called `minerva` that can be used to generate responses, given some parameter values and experimental variables:

```
1 minerva=function(L,crit,decay,n.features,alpha,p,n.study,n.
  targets,n.test){
2   study.feats=init(n.features*n.study,p) # features of study list
3   study=matrix(study.feats,n.features,n.study) # study list
4   image.feats=rbinom(n.features*n.study,1,L) # feature of image
5   image=matrix(image.feats,n.features,n.study)*study # image
6   image=rbinom(n.features*n.study,1,1-decay)*image # decay
7   targets=study[,1:n.targets] # use first study items (arbitrary
  )
8   # calculate number of distractors, and generate them
9   n.distractors=(n.test-n.targets) # how many distractors?
10  dist.feats=init(n.features*n.distractors,p) # features
11  distractors=matrix(dist.feats,n.features,n.distractors) # set
12  test=cbind(targets,distractors) # create test set
13  S=matrix(NA,n.study,n.test) # create similarity matrix
14  for(i in 1:n.test){ # loop over test items
15    for(j in 1:n.study){ # ...and study items
16      calc.n=sum(test[,i]!=0 | image[,j]!=0) # nonzero features
17      S[j,i]=sum(test[,i]*image[,j])/calc.n # similarity
```

```

18 }
19 }
20 A=S^alpha # calculate activation
21 out=apply(A,2,sum) # calculate intensity
22 as.numeric(out>crit) # compare to criterion
23 }

```

The first line of code declares the function, which requires the learning rate L (i.e., L), the criterion value $crit$ (i.e., C), the decay rate $decay$ (i.e., δ), the number of features $n.features$ (i.e., η), the exponent of the similarity matrix $alpha$ (which is commonly set to $\alpha = 3$, as in Eq. (3.2)), and the feature probability vector p (which is usually set to $p = \{1, 1, 1\}/3$, so that the features are equally likely). In addition, the `minerva` function requires some choices about the details of the experiment, such as the number of items presented at study (i.e., `n.study`), the number of items in the test list that were on the study list (i.e., `n.targets`), and the total number of items in the test set (i.e., `n.test`). Lines 2 and 3 create the features of the study list, and then arrange those features into a matrix where the columns correspond to the items, and the rows correspond to the features. Lines 4–6 detail the construction of the episodic memory matrix. First, Line 4 generates a vector of Boolean variables declaring whether or not the features of the study list should be encoded. The vector in Line 4 is rearranged into a matrix and multiplied by the study list, creating a new matrix that represents the episodic image: some features of the study list will appear within the image matrix at the same location as in the study list matrix, whereas some values within the image matrix will be zero, indicating no features were encoded. Finally, the quality of the image matrix is further deprecated by multiplying the matrix by another Boolean matrix representing the feature decay process. During this multiplication, some features that were correctly encoded in Line 5 will be reset to zero, eliminating those features from contributing to the recognition decision at test. Line 7 declares that the first `n.targets` items will be selected from the study list to serve as the targets in the test list. Because we are simulating a model and not using a human subject, the choice of selecting target items from the study list is completely arbitrary. Lines 9–11 create the set of distractors to be used in the test set, and Line 12 creates the final test set by combining targets and distractors. Again, because we are simulating the model, the arrangement of targets and distractors is inconsequential to the pattern of responses we will simulate.

The next step is to calculate the similarity matrix S , which is performed in Lines 13–19. To do this, we follow Eq. (3.1) by looping through the set of test items and the set of items in the episodic image. The first step is to calculate ϵ_i in Line 16 to determine how many features are nonzero in either the current test item or the current episodic image item (i.e., determined by i or j in the double `for` loop). Finally, Line 17 performed the summation in Eq. (3.1) through matrix multiplication. Line 20 calculates the activation values by cubing the similarity matrix shown in Eq. (3.2), and Line 21 calculates intensity according to Eq. (3.3). Finally, to make a response, the model compares the intensity of each test item to the criterion variable `crit`: if the intensity for Item i is larger than the criterion, an

“old” response is given, whereas if the intensity value is smaller than the criterion, a “new” response is given.¹

The Minerva 2 model is relatively simple to set up and simulate data from, and as a consequence, it serves as an interesting running example on which we can apply likelihood-free techniques to illustrate the utility of these methods. Despite Minerva 2’s simplicity, to our knowledge researchers have not yet taken full advantage of Bayesian hierarchical methods in fitting the model because it is simulation-based. In the following section, we will describe how likelihood-free techniques can be used to fit this model to data. We first fit the model to simulated data to demonstrate the methods’ ability to recover the model parameters. Then, we use these techniques to fit the model to recognition memory data from a real-world experiment.

3.3 Simulation Study: Recovering the Posterior Distribution

Although the Minerva 2 model can be fit to a variety of data, for our simulation study, we focus on the recognition memory task. Recognition memory data are perfect candidates for illustrating the likelihood-free approach for two reasons. First, the number of measurements (i.e., hit and false alarm rates) from each subject is generally small, and so simulating the model to match the observed data is not very computationally costly. Second, the hit and false alarm rates are discrete: measures are incremented in steps of $1/n$, where n is either the number of targets (for the hit rate) or distractors (for the false alarm rate). This means that when using methods such as PDA, the error introduced in the estimation of the posterior distributions will be minimized, as a kernel density function is not needed to approximate the shape of the probability density function [38].

3.3.1 Generating the Data

For this simulated experiment, we assumed that the test list consisted of 40 items total, 20 of which were targets (i.e., words on the previously studied list) and 20 of which were distractors (i.e., words not on the previously studied list). The study list consisted of 20 items, all of which were presented during the test phase. Each subject completed four conditions of the recognition memory task. Hence, the simulated data consist of four hit rates and four false alarm rates for each subject. The larger data set provides an opportunity for the posterior distribution to be different from the prior distribution, thus creating a greater constraint on the model. This increased stringency allows us to appreciate the quality of the likelihood approximation used by the three methods below.

¹The responses are arbitrarily coded as either a one for an “old” response, or a zero for a “new” response.

To simulate data from the model, we set the learning rate $L = 0.5$, the criterion $C = 0.10$, the number of features $\eta = 30$, and the decay rate $\delta = 0$. We set the α parameter to 3 and the probability of the features taking on the values $\{-1, 0, 1\}$ to be $\{1/3, 1/3, 1/3\}$, respectively. When simulating the model for four subjects, the following hit and false alarm rates were obtained:

$$\text{Hit Rates : } \{0.40, 0.75, 0.75, 0.60\}$$

$$\text{False Alarm Rates : } \{0.05, 0, 0, 0\}$$

This set of data will be used in the posterior recovery test below, and they are important as changes in the data above may result in changes to the posterior estimates obtained below.

For our posterior recovery test, we only estimated L , C , and η , because the analytic expressions derived in Sheu [37] did not consider the effects of the decay parameter δ . Hence, because analytic expressions for this expanded version of the model are unavailable, we assumed that δ was known to facilitate a comparison across the three methods. All parameters were equal across all four conditions of the experiment.

3.3.2 Fitting the Data

To illustrate the likelihood-free approach, we fit the model to the simulated data in three different ways. The first approach is the kernel-based ABC algorithm, which relies on summary statistics to approximate the likelihood. The second approach is the PDA method [38], which constructs an approximation to the missing likelihood via pure simulation. The third approach relies on analytic expressions derived by Sheu [37], which rely on asymptotic assumptions about the distribution of hit and false alarm rates conditional on a set of model parameters. As we will discuss in detail below, by “analytic” we mean that the approximation of the likelihood has a functional form, but we do not necessarily mean that the expressions are perfectly accurate. In order to obtain analytic expressions Sheu made some simplifying assumptions about the asymptotic properties of the distribution of echo intensities. While these assumptions are reasonable for infinitely long lists, their validity when applied to data with finite limitations has not yet been tested.

Each of the three methods is unique in the way they approximate the posterior distribution. However, when sampling from the posterior distribution, another set of algorithms are required to perturb the proposals throughout the parameter space so that an accurate posterior estimate can be achieved. To maintain consistency across the three methods, we applied an MCMC algorithm (see Chap. 2) with identical settings to obtain samples from the posterior distribution using each approximation method. Note that while the MCMC algorithm is identical, because each approximation method is different, we cannot necessarily expect that each

procedure will result in identical posterior estimates; in fact, it is this comparison of posteriors that we will use to evaluate the quality of each approximation method.

While we cannot reproduce the code for the entire MCMC algorithm here, we encourage the reader to consult the online materials for versions of each method implemented in R. Again, all methods use an identical MCMC sampler to perturb proposals within the parameter space, yet have different methods for evaluating the quality of the proposal. At their core, all methods invoke a function that specifies the log likelihood of the data (i.e., the variable `data`), given a proposal parameter value (i.e., the variable `x`). The general form of the log likelihood function looks like the following block of code:

```

1 log.dens.like=function(x,data){
2   L=x[1]; crit=x[2]; feat=x[3]; # redeclare parameters
3   feat=round(feat) # round the number of features
4   if(L<=1 & L>=0 & feat>=2){ # test parameter boundaries
5     ### insert specific approximation method here
6     ### producing a variable called 'out'
7     if(is.na(out))out=-Inf # test for plausibility
8   } else { # if boundary test fails...
9     out=-Inf # reject the proposal
10  }
11  out # return the final log likelihood value
12 }

```

Line 1 through Line 12 declare the log likelihood function in R. As you can see, the function requires two inputs—the parameter proposal `x` and the set of data `data`. For convenience, Line 2 transforms the elements of the proposal vector `x` into the learning rate variable `L`, the criterion variable `crit`, and the number of features variable `feat`. Line 3 rounds the feature variable into something discrete, so that it can be used in the `minerva` function to construct the episodic image (i.e., only discrete values can be used as the dimensions of a matrix). Next, Line 4 tests to see whether or not each parameter value is within a plausible range. Statistically speaking, this line is not completely necessary as these restrictions will be specified in our priors. However, algorithmically, the `minerva` function above will crash if these restrictions are not in place. Line 4 is connected to Lines 8–10 as a condition statement. That means that if a parameter value is outside the range of plausible values, the final log likelihood value `out` will be set to $-\infty$ or `-Inf` in Line 9. Lines 5–6 represent the location where lines of code can be inserted to implement each approximation method, which we will discuss below. Line 7 is a final test to evaluate whether or not the implementation method produced a log density value that is plausible. As we assume that NA values correspond to values of the parameters that are implausible, this line is the final “catch” to rid our posteriors of invalid samples. Hence, it is very important to ensure that the approximation method—and more specifically the model data generation code—is as robust as possible. Finally, Line 11 produces the likelihood of the data, given a set of parameters, on the log scale.

With the generic wrapper function described, we can now turn to the specific implementation details that can be slotted into Lines 5–6 above.

3.3.2.1 KABC

As we discussed in Chap. 2, kernel-based ABC (KABC) relies on a kernel to compare the simulated data to the data that were observed. We chose a Gaussian kernel with standard deviation $\delta_{\text{ABC}} = 0.03$. This particular kernel worked well, giving accurate posterior estimations while still allowing the chains in the MCMC sampling algorithm to mix properly.² For each proposed parameter value, we simulated the model under the proposal four times to reflect the number of observed data points. Doing this allows for straightforward comparison of the simulated data to the observed data, although other choices are possible [44].

To implement the KABC algorithm, we adapted code in R based on Fig. 2.3 that would work generically across the three approximation methods. We again encourage the reader to see the scripts associated with each method, as we cannot reproduce them here. Instead, the specifics of implementing the KABC approximation can be seen in the following block of code:

```

1 mach=matrix(NA,S,n.test) # declare a matrix
2 for(i in 1:S){ # loop over S subjects (i.e., S=4 here)
3 # simulate data from the model with the following settings:
4 # unknown parameters: L, crit, feat
5 # known parameters: decay, t.alpha, p
6 # known experimental variables: n.study, n.targets, n.test
7 mach[i,]=minerva(L,crit,decay=0,feat,t.alpha=3,p=c(1,1,1)*1/3,
8 n.study,n.targets,n.test)
9 }
10 # calculate summary statistics for the simulated data
11 mach.hr=apply(mach[,1:n.targets],1,sum)/sigs # get HR
12 mach.fa=apply(mach[, (n.targets+1):n.test],1,sum)/noise # get
    FAR
13 # evaluate how close the simulated and observed data are
14 out.hr=sum(log(dnorm(mean(data$hr-mach.hr),0,.03))) # compare HR
15 out.fa=sum(log(dnorm(mean(data$fa-mach.fa),0,.03))) # compare
    FAR
16 out=out.hr+out.fa # calculate final 'out' variable

```

First, Line 1 declares a storage object. Line 2–9 perform a simulation using our `minerva` function from above by looping over each subject, generating data using a set of unknowns (i.e., the three parameters we are estimating) and knowns (i.e., the fixed parameters and the experimental setup), and storing the simulated responses with the matrix `mach`. Once the data have been simulated, the next step is to calculate some summary statistics. For our purposes, the set of summary statistics we chose were the hit and false alarm rates, which are calculated in Lines 11 and 12, respectively. Next, we evaluate how closely the summary statistics for the simulated data are to the observed data by using a Euclidean distance and a Gaussian kernel (as in Chap. 2) with mean 0 and standard deviation 0.03 in this case. As we are computing the log likelihood value, we can simply sum up the log-transformed

²In our simulations, we tested a few different values of δ_{ABC} until we arrived at the smallest value that still produced good mixing behavior across the chains.

likelihoods, resulting in the variables `out.hr` and `out.fa`. The final step is to obtain the log likelihood of all the data, which is obtained by summing up `out.hr` and `out.fa` in Line 16. This last line produces the variable `out` that can be used in the generic `log.dens.like` function above.

3.3.2.2 PDA

Following the details described in the previous chapter, we constructed an approximation of the joint probability density functions for hit and false alarm rates by simulating the model 1000 times for each parameter proposal. From this bivariate distribution we calculated the probability of observing the data under this parameter proposal: we evaluated the density of the constructed distribution at the location of each observed hit and false alarm rate. To implement this method, we can use the following block of code:

```

1 mach=matrix(NA,rep,n.test) # declare a matrix
2 for(i in 1:rep){ # simulate 'rep' times (i.e., rep=1000 here)
3 # simulate data from the model with the following settings:
4 # unknown parameters: L, crit, feat
5 # known parameters: decay, t.alpha, p
6 # known experimental variables: n.study, n.targets, n.test
7 mach[i,]=minerva(L,crit,decay=0,feat,t.alpha=3,p=c(1,1,1)*1/3,
8 n.study,n.targets,n.test)
9 }
10 # calculate the hit and false alarm rates for each simulation
11 mach.hr=apply(mach[,1:n.targets],1,sum)/sigs
12 mach.fa=apply(mach[, (n.targets+1):n.test],1,sum)/noise
13 pdf=numeric(S) # declare a storage object
14 for(j in 1:S){ # loop over subjects
15 # determine the joint probability of obtaining each observed
16 # data point, given the distribution of simulated data
17 pdf[j]=mean(mach.hr==data$hr[,j] & mach.fa==data$fa[j])
18 if(is.na(pdf[j])==T)pdf[j]=0 # test to ensure no NA values
19 }
20 out=sum(log(unlist(pdf))) # sum up the log likelihood values

```

The PDA code is similar to the KABC code but there are some important differences. First in Line 1, the matrix `mach` is constructed to have `rep` rows and `n.test` columns. The difference here is that `rep` will be large relative to `S` from the above KABC code. Because `S` is just the number of subjects, the KABC code will generate data of the same size as the data that were observed. By contrast, the PDA code will construct a full distribution over the space of possible hit and false alarm rates. Lines 2–9 simulate the Minerva 2 model `rep` times and place the elements within the matrix `mach`. Lines 11 and 12 next complete the hit and false alarm rates for the `rep` simulations. The next step is to construct the simulated PDF. First, a `pdf` variable is constructed to contain all `rep` densities. Line 17 computes the joint probability that the observed hit and false alarm rates (i.e., `data$hr` and `data$fa`, respectively) match the hit and false alarm rates from the simulation above. To do this, we calculate the number of times the two elements match at the

same time and then divide by the total number of observations. Another convenient way to do this is to simply take the mean of the boolean vector that performs the match comparison, as in Line 17. While this will compute the joint probability for a single data point, we must repeat this process for all of the observed data, and so the loop in Lines 14–19 is designed to carry this operation out. The final step is to check for NA values (i.e., Line 18) and construct the variable `out` as we did above.

3.3.2.3 Analytic Expressions

Sheu [37] derived analytic expressions for the Minerva 2 model. These expressions are based on asymptotic assumptions, and so they describe, in the limit, the mean and variance of the lure and target activation distributions. The lure and target activation distributions are assumed to be normally distributed. In our description of Minerva 2 above, we assumed that the probabilities of a feature taking on one of the values in the set $\{-1, 0, 1\}$ were equal. This is a simplifying assumption that is regularly used in practice. More generally, we can denote the probability of a feature taking on these three values as r , q , and p , respectively, where $r + q + p = 1$. In a typical application, the probability that the features take on a value other than zero is equally likely, such that $r = p$, and so $q = 1 - 2p$. Sheu [37] maintained this assumption for simplicity. Using this constraint and our notation from above, when a target is presented, the mean and variance of the activation A^+ for targets are

$$\begin{aligned} E[A^+] &\approx L^3 + 3L^2(1-L)\mu_{\eta,p}, \text{ and} \\ \text{Var}[A^+] &\approx 9L^5(1-L)\mu_{\eta,p}, \end{aligned}$$

where

$$\mu_{\eta,p} = \sum_{k=1}^{\eta} \binom{1}{k} \binom{\eta}{\eta-k} (1-2p)^{\eta-k} (2p)^k.$$

Although the distribution of activation for targets is easy to describe, the distribution of activation for distractors A^- is more difficult. The mean and variances of A^- are

$$\begin{aligned} E[A^-] &= 0, \text{ and} \\ \text{Var}[A^-] &= \sum_{S^-=s} s^3 p_{S^-}(s), \end{aligned}$$

where $p_{S^-}(s)$ is the probability mass function of the distribution of similarity for distractors.

To sum across the similarity distribution for distractors, we must first calculate the echo intensity variances for distractors. Letting k be an index over all of the non-zero elements of a trace such that $0 \leq k \leq \eta$, and h be an index such that

$-k \leq h \leq k$, the probability function of the similarities of the i th item in the episodic memory matrix is

$$p_{S_i^-}(h/k) = \binom{\eta}{\eta-k} [(1-2p)(1-2Lp)]^{\eta-k} \times \sum_{u+v=w=h} \binom{k}{uvw} (2Lp^2)^{u+w} [2p(1+L-4pL)]^v, \quad (3.4)$$

where u , v , and w are non-negative integers constrained such that $0 \leq u, v, w \leq k$ and $u + v + w = k$. Equation (3.4) is difficult to calculate efficiently, so Sheu constructed a set of recursive equations that are easier to evaluate.

When the means and variances of both the lure and target activation distributions have been evaluated, one can compute the probability of obtaining a hit and false alarm by integrating both normal distributions from the criterion parameter C to infinity. Specifically, the hit rate H and the false alarm rate FA predicted by the model are

$$P(H|\theta) = 1 - \Phi\left(C|E[A^+], \sqrt{\text{Var}[A^+]}\right) \text{ and}$$

$$P(\text{FA}|\theta) = 1 - \Phi\left(C|E[A^-], \sqrt{\text{Var}[A^-]}\right),$$

where $\Phi(x|a, b)$ is the normal cumulative density function at x with mean parameter a and standard deviation b . To connect these probabilities to the likelihood function, we invert the probability structure by way of multiplication as we saw in Chap. 1. We denote the number of hits for the i th subject as $O_{i,T}$ (i.e., T for targets) and the number of false alarms by $O_{i,D}$ (i.e., D for distractors), and we assume that the number of hits $O_{i,T}$ and false alarms $O_{i,D}$ arise from binomial distributions with the number of trials equal to the number of targets $N_{i,T}$ and distractors $N_{i,D}$, respectively. We can estimate the probability that the model makes an “old” response to both targets (i.e., the hit rate) and distractors (i.e., the false alarm rate) by multiplying the two probabilities together, such that

$$L(\theta|Y) = \prod_{i=1}^S \text{Bin}(O_{i,T} | N_{i,T}, P(H|\theta)) \text{Bin}(O_{i,D} | N_{i,D}, P(\text{FA}|\theta)), \quad (3.5)$$

where Y is the observed data $Y = \{O_{1:S,T}, O_{1:S,D}\}$, $S = 4$ is the number of observations, and $\theta = \{C, \eta, L\}$, and $\text{Bin}(x|n, p)$ is the density of the binomial distribution, given by

$$\text{Bin}(x|n, p) = \binom{n}{x} p^x (1-p)^{n-x}.$$

To form the analytic approximation of the likelihood, we followed Sheu [37] and programmed a routine in R to evaluate the equations listed above. The code for implementing the model is provided in the software release with this book. To sample from the posterior distribution, we can insert the following block of code into the likelihood wrapper as we did in the KABC and PDA sections above:

```

1 # unknown parameters: L, crit, feat
2 # known parameters: decay, t.alpha, p
3 # known experimental variables: n.study, n.targets, n.test
4 # calculate the mean and variances from Sheu (1992)
5 temp=minerva2_analytic(L,feat,p)
6 # calculate the hit rate predicted by the model
7 temp.hr=1-pnorm(crit,temp$mean.signal,sqrt(temp$var.signal))
8 # calculate the false alarm rate predicted by the model
9 temp.fa=1-pnorm(crit,temp$mean.noise,sqrt(temp$var.noise))
10 # compute the final joint probability of the data
11 out.hr=sum(log(dbinom(data$hr*sigs,sigs,temp.hr))) # HR
12 out.fa=sum(log(dbinom(data$fa*noise,noise,temp.fa))) # FAR
13 # compute the final joint probability of the data
14 out=out.hr+out.fa

```

As before, the unknown parameters, known parameters, and experimental variables are listed in Lines 1–3. The first step is to pass the parameters L and η to the function `minerva2_analytic` which computes the mean and variances of the signal and noise distributions. Notice that this function does not depend on the experimental variables, as they reflect the long-term representations used in the model to make recognition decisions. This assumption may be problematic as Minerva 2 is a global matching model and depends on the properties of the study set. Also note that the decay term δ is not passed to the `minerva2_analytic` function as the likelihood approximation developed by Sheu [37] only considers the predictions of the model under the constraint that $\delta = 0$. Next, Lines 6–9 compute the hit and false alarm rates by assuming normal representations for both target and lure distributions (i.e., as in [37]). Here, the parameter C (i.e., the variable `crit`) is used to specify the boundary point of the integral of the normal distributions, which provides the hit and false alarm rates predicted by the model. Finally, Lines 10–14 compute the joint probability of the observed data by adding together the log densities from the binomial distribution.

3.3.3 Results

For each method, we used a generic MCMC algorithm to sample from the posterior distribution. We used a Gaussian kernel for each dimension in the parameter space and set the standard deviations to 0.1, 0.05, and 3 for L , C , and η , respectively. We ran the sampler with 24 chains for 2000 iterations, and used a burn-in period of 200 iterations. We then thinned the chains by discarding every other sample. Hence, each method produced 21,624 samples of the joint posterior distribution.

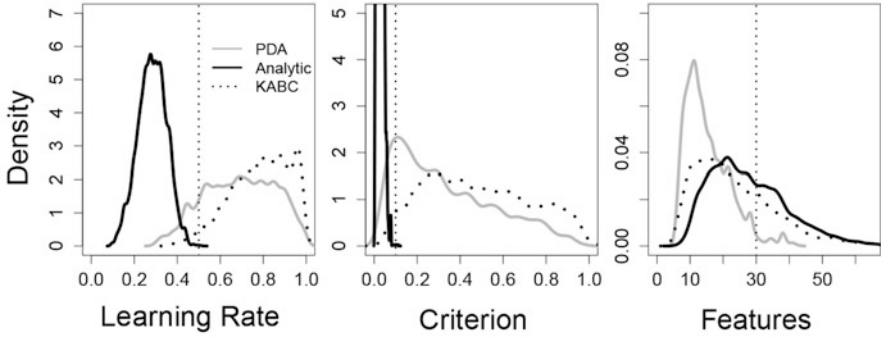


Fig. 3.1 Marginal posterior distributions for the three Minerva 2 parameters. The left, middle, and right panels show the estimated marginal posterior distributions for the learning rate L , the criterion C , and the number of features η , respectively. In each panel, three estimates are shown, one corresponding to each method used to form the approximation: PDA (gray), analytic expressions (solid black), and KABC (dotted black). In each panel, the true value of the parameter used to generate the data is shown as the vertical dashed line

Figure 3.1 shows the estimated marginal posterior distributions for each of the model parameters. The left panel shows the estimate for the learning rate L , the middle panel shows the estimate for the criterion C , and the right panel shows the estimate for the number of features η . Within each panel, the estimate obtained using each of the three methods is shown: PDA (gray), analytic expressions (solid black), and KABC (dotted black). In each panel, the true value of the parameter used to generate the data is shown as the dashed vertical line. Comparing across the three methods, Fig. 3.1 shows that the estimates obtained using PDA and KABC are more similar to one another than they are to the estimates obtained using the analytic expressions.

In particular, the estimate from the analytic expressions for the criterion parameter is highly peaked around the value 0.03 and has considerably less variance compared to the other likelihood-free methods. However, the estimated posterior is so heavily concentrated that it does not overlap with the value of the parameter used to generate the data. Similarly, the estimated posterior for the learning rate parameter using the analytic expressions also misses the true value of the parameter used to generate the data. Yet, the estimates for the learning rate parameter using KABC and PDA do overlap with the true value, and are somewhat similar to one another.

Because the true likelihood function has not been derived for Minerva 2, we cannot say for sure which of the three estimates is correct. We suspect that the inaccuracy of the posterior distributions in Fig. 3.1 reflects the effects of the normality assumption used in Sheu [37]. Although assuming that the target and lure activation distributions are normally distributed is convenient for deriving an approximation for the likelihood function, the activation distributions produced by Minerva 2 are only normal in the limit as the number of features in the trace

vector increases and the number of traces in the episodic memory matrix increases. Conventional values for Minerva 2 for number of features and number of traces are around 20, and so it is possible that the approximation is less accurate under these conditions. However, because the estimates obtained from the analytic expressions miss the true value used to generate the data, we suspect that this approximation is less accurate than either the PDA or KABC methods.

The results in Fig. 3.1 also show some small differences between PDA and KABC. We suspect that the reason for the differences is due to the kernel-based approximation, which introduces some approximation error over and above the error due to Monte Carlo sampling. By contrast, the PDA method removes this approximation error, but also has the some error associated with the construction of the simulated PDF. Again, as we do not know what the true posterior distribution is, we cannot properly evaluate which estimate is more accurate (but see Chap. 4 for some validation examples).

In addition to the marginal distributions shown in Fig. 3.1, we can also examine the estimated joint posterior distributions. Occasionally, these posterior distributions reveal interesting tradeoffs in the model parameters that would otherwise be difficult to appreciate [38, 44]. Figure 3.2 shows the estimated joint posterior distribution obtained using the PDA method for each pairwise combination of the three model parameters. The left panel plots the criterion C against the learning rate L , the middle panel plots the number of features η against L , and the right panel plots η against C . Figure 3.2 reveals an interesting curvilinear pattern in the posteriors, especially in the left and right panels. Although a detailed analysis of Minerva 2 is outside the scope of this tutorial chapter, we can provide some intuition behind the tradeoff between the learning rate L and the number of features η . Recall that the memory traces are formed by a fixed number of features representing the items in the study list. After an item is presented for study, features of that item are copied into the episodic memory matrix with probability L . As L increases, the probability

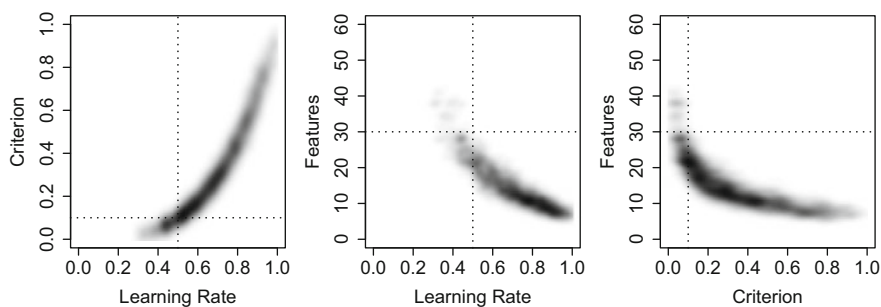


Fig. 3.2 Joint posterior distributions for the three Minerva 2 parameters. The estimated joint posterior distribution obtained using the PDA method for each pairwise combination of the three model parameters are shown: the criterion C against the learning rate L (left), the number of features η against L (middle), and η against C (right). The true values of the parameters used to generate the data are shown as the dashed lines

of copying a given feature increases, making recognition performance better at test. Hence, when the learning rate is low, Minerva 2 needs more features of the item to accurately copy enough features to maintain the same accuracy observed in the data. A similar pattern of tradeoffs exists in the Retrieving Effectively from Memory (REM) [44, 83, 92] model.

We can learn a good deal about model constraint, flexibility, and identifiability by close examination of the model parameters. For example, in typical applications, we assume a diffuse prior on the model parameters. After the posterior distribution has been estimated, we can compare the spread of the prior distribution relative to that of the posterior. If a significant discrepancy is observed, where the posterior distribution has smaller variance than the prior distribution, we can conclude that the data are constraining the model, because the data have provided evidence that reduces our uncertainty about the parameter values. Hence, the spread of the joint posteriors in Fig. 3.2 suggests that the parameters are well identified, and we learn a lot from our data. In addition, the parameters are highly correlated and trade off against each other, a finding that is often observed in computational psychological models [38, 44, 57]. Finally, the joint posteriors reveal reasonably good accuracy with respect to recovering the true parameter values (also see Fig. 3.1), which further suggests the model is well identified.

In general, if we were to assess whether or not the posterior estimates were accurate, we would need the true likelihood function so that we could compare the estimates obtained by the approximation methods. However, as we don't currently have the likelihood function for the Minerva 2 model, we must instead compare the estimated posteriors to the true value used to generate the data. Unfortunately, the comparison is not as simple as evaluating the density of the true value of the parameter within the posterior distribution. In fact, there is nothing that guarantees an estimated posterior distribution will center around a true parameter value in these types of simulation studies. All we can assess is whether or not the true value is contained within the posterior. For KABC and PDA, all of the true values are contained somewhere within the posterior. However, for the analytic expressions, the criterion parameter and the learning rate do not contain the true value.

We also measured the total computational time required to complete each simulation. The PDA method took 10 min and 2 s, the analytic method took 6 min and 26 s, and the KABC method took 6 min and 58 s. The PDA and KABC method both require simulations of the model. For optimization purposes, we programmed Minerva 2 in the C language. The analytic expressions do not require model simulation, but they do require a number of calculations [37]. We used R to make the analytic calculations and did not export the code to C because the R version was relatively fast. To perform each simulation, we parallelized the computation across 8 cores on a Mac Pro desktop computer with a 3.7 GHz processor. The computation times reveal an interesting tradeoff. First, the fastest results are obtained using the analytic expressions, and the slowest using the PDA method. However, we believe that the analytic expressions are also the most inaccurate, and the PDA method is the most accurate as no error terms corrupt the estimates. However, the differences between the PDA and KABC estimates are not large, which suggests that the KABC algorithm may be a suitable approximation for more intensive model fits.

3.4 Real-World Application: Dennis et al. [4]

Our next exercise fits a hierarchical version of Minerva 2 to data from Dennis et al. [4]. In this experiment Dennis et al. [4], had people perform a recognition memory task that manipulated list length and the presence or absence of an additional filler task between the study and test phase of the experiment. In addition, they implemented a number of controls to eliminate confounds present in traditional list length designs. The first of these confounds is an unequal retention interval across short and long list length conditions, which they controlled by keeping the time between the onset of the study list and the onset of the test list the same in both conditions. The second is the possibility of a decrease in attention which might occur over the presentation of the study list. They controlled this possibility by only testing items from the beginning of the study list in both short and long list conditions.

Dennis et al. [4] found that the list length effect depended on the presence of the filler task. Specifically, when no filler task followed the study list, recognition performance was better for short lists than for long lists. However, when a filler task was present there was no list length effect. We chose to fit the model to this particular dataset because the two independent variables, list length and retention interval, are both variables for which Minerva 2 makes predictions.

3.4.1 The Model

To fit Minerva 2 to the data from Dennis et al. [4] we must first recall the roles that each parameter in the model plays and their relationships to the experimental design.³ We discussed already how Minerva 2, as other global matching models, accounts for effects of list length as a function of the number of trace vectors in the memory matrix. Specifically, the matching process comparing an item from the test list to the episodic image involves a comparison to every item from the study list. Next, we must consider how the effects of the filler task might influence memory performance in each task, and how the model might account for these changes across tasks. We assumed that the effects of the filler task can be explained by the decay rate parameter δ . We assume that, in the presence of a filler task, the individual traces in memory decay with probability δ , an assumption that is consistent with the original implementation of the model [79]. Recall that if a feature in a trace decays that it reverts to a zero in the episodic image. As a result, increases in δ produce more memory decay, resulting in lower discriminability.

To implement this mechanism we define a binary indicator F_j to designate the parameters for a given condition j . We let F_j represent the condition in which the

³Dennis et al. [4] also used words of different frequency (high and low) to construct their study and test lists. Because Minerva 2 lacks a mechanism for explaining word frequency effects in recognition memory, for the purposes of this demonstration we collapsed across both word frequency classes to produce a single hit and false alarm rate for each experimental condition.

additional filler activity was either absent ($F_j = 0$) or present ($F_j = 1$). We can then write the parameter vector $\theta_{i,j}$ for the i th subject in the j th condition as

$$\theta_{i,j} = \{L_i, \delta_i F_j, \eta_i, C_i^{(j)}\}. \quad (3.6)$$

Thus, in conditions with no filler task ($F_j = 0$), the decay parameter $\delta_i = 0$, but when a filler task is present ($F_j = 1$), $0 \leq \delta_i \leq 1$. The assumption setting the decay rate is zero when no filler task is present is an arbitrary one, as the other model parameters should scale accordingly. The important difference here is the value of the decay rate δ_i relative to zero.

For the i th subject in the j th condition, we denote the number of hits as $O_{i,j,T}$ and the number of false alarms by $O_{i,j,D}$. The number of hits $O_{i,j,T}$ and false alarms $O_{i,j,D}$ arise from binomial distributions with the number of trials equal to the number of targets $N_{i,j,T}$ and distractors $N_{i,j,D}$, respectively. We can estimate the probability that the model makes an ‘‘old’’ response to both targets (i.e., the hit rate) and distractors (i.e., the false alarm rate) by simply simulating the model many times and tabulating the responses under the different stimulus types. These values give us the probability of a hit $P(H | \theta_{i,j})$ and false alarm $P(FA | \theta_{i,j})$. Letting the observed data

$$\begin{aligned} Y = & \{\{O_{1,1,T}, O_{1,1,D}\}, \{O_{1,2,T}, O_{1,2,D}\}, \dots, \{O_{1,J,T}, O_{1,J,D}\}, \\ & \{O_{2,1,T}, O_{2,1,D}\}, \{O_{2,2,T}, O_{2,2,D}\}, \dots, \{O_{2,J,T}, O_{2,J,D}\}, \\ & \dots, \\ & \{O_{S,1,T}, O_{S,1,D}\}, \{O_{S,2,T}, O_{S,2,D}\}, \dots, \{O_{S,J,T}, O_{S,J,D}\}\} \end{aligned}$$

for J conditions and S subjects, the likelihood function for Y is

$$L(\theta | Y) = \prod_{i=1}^S \prod_{j=1}^J \text{Bin}(O_{i,j,T} | N_{i,j,T}, P(H | \theta_{i,j})) \text{Bin}(O_{i,j,D} | N_{i,j,D}, P(FA | \theta_{i,j})), \quad (3.7)$$

where $S = 48$ is the number of subjects, and $J = 4$ is the number of conditions.

To complete the model we select priors for each of the parameters. Because the parameters L_i and δ_i represent probabilities, they are both restricted to be between zero and one. We therefore assumed that each of the individual-level parameters δ_i and L_i have truncated normal priors bounded by zero and one, or

$$\begin{aligned} L_i & \sim \mathcal{TN}(\omega_L, \xi_L, 0, 1), \text{ and} \\ \delta_i & \sim \mathcal{TN}(\omega_\delta, \xi_\delta, 0, 1), \end{aligned}$$

where $\mathcal{TN}(a, b, c, d)$ denotes the truncated normal distribution with mean a , standard deviation b , lower bound c , and upper bound d . Because the number of features parameter η_i can only take on integer values, we used a discretized version

of the truncated normal distribution with mean parameter ω_η , standard deviation parameter ξ_η , and with a lower bound of two so that

$$\eta_i \sim \mathcal{DTN}(\omega_\eta, \xi_\eta, 2, \infty),$$

where $\mathcal{DTN}(a, b, c, d)$ denotes the discretized truncated normal distribution with mean a , standard deviation b , lower bound c , and upper bound d . Choosing a lower bound of two restricted each item to have at least two features.⁴

Unlike current Bayesian recognition memory models [83, 85], Minerva 2 and the other global matching models use different criteria for an “old” response for different experimental conditions. This is because changes in Minerva 2 parameters, such as the learning rate and decay rate, affect the mean of the target activation distribution without changing the mean of the distractor activation distribution (although the variances of these distributions are different). To illustrate why a single criterion is insufficient, consider two experimental conditions with different retention intervals, or the length of time between study and test phases of the experiment. The longer retention interval condition will have a higher decay rate, which will decrease the mean of the target activation distribution. If the decision criterion is fixed for long and short retention intervals, the lower mean target activation will result in lower hit rates, but the false alarm rate will not be affected. If the decision criterion is reduced in the long retention condition, the hit rate will be lower and the false alarm rate will be higher than in the short retention condition, which is consistent with experimental findings. Allowing for different decision criteria across conditions also allows for unbiased responding in each condition.

To ensure that the model can fit the data, we assumed a separate criterion parameter $C_i^{(j)}$ for each of the $J = 4$ conditions. This is not without precedent, as a similar approach was used by Clark and Shiffrin [93] to fit Minerva 2 to their recognition data. For each individual criterion we specified a normal prior distribution, so that

$$C_i^{(j)} \sim \mathcal{N}\left(\omega_C^{(k)}, \xi_C^{(k)}\right).$$

Because Minerva 2 has never been fit hierarchically to data, we have no information about the likely ranges of the hyperparameters. As a consequence, we used noninformative priors for the hyperparameters to reflect this uncertainty. First, for the mean parameters ω_L and ω_δ , we chose a uniform distribution that put equal density on all values in the interval (0,1), so that

$$\omega_L \sim \text{Beta}(1, 1), \text{ and}$$

$$\omega_\delta \sim \text{Beta}(1, 1).$$

⁴After fitting the model, we noticed that no marginal distribution for η_i went below four, so while our choice was made out of convenience, it had little effect on the posterior estimates.

To choose priors for the criteria and number of features, we examined the model predictions under a variety of different choices for $\{\eta, C\}$. We found that values of C ranging from 0 to 0.5 with the number of feature ranging from 10 to 20 produced data that one might expect from a typical recognition memory task. Thus, we settled on mildly informative priors for these parameters, given by

$$\begin{aligned}\omega_C^{(k)} &\sim \mathcal{N}(0.05, 1), \text{ and} \\ \omega_\eta &\sim \mathcal{TN}(40, 15, 2, \infty).\end{aligned}$$

For the standard deviation hyperparameters $\xi = \{\xi_L, \xi_\delta, \xi_C^{(k)}, \xi_\eta\}$, we used a common, mildly informative priors, so that

$$\xi \sim \Gamma(1, 1),$$

We chose these priors because we expected only a moderate degree of variability in the individual-level parameters, and the $\Gamma(1, 1)$ distribution covered a sufficiently large range.

Figure 3.3 shows a graphical diagram for this hierarchical model. These types of diagrams are often very useful for illustrating how the parameters in the model (white nodes) are connected via arrows to the observed data (gray nodes) [6, 12, 17]. When the variables are discrete they are shown as square nodes, whereas when the variables are continuous they are shown as circular nodes. A double bordered variable indicates that the quantity is deterministic, not stochastic, and computed from other variables. For example, the node corresponding to $\theta_{i,j}$ is double bordered because it is always determined by evaluating Eq. (3.6). Finally, ‘‘plates’’ show how vector-valued variables are interconnected. For example, the node for the parameter ω_L is not within the plates, which indicates that this parameter is fixed across both subjects and conditions, whereas there are separate L_i nodes for every subject, and separate $C_i^{(j)}$ nodes for each subject and condition.

3.4.2 Results

To fit the hierarchical model we used the PDA method [56] embedded within the Gibbs ABC algorithm [70]. We generated proposals using DE-MCMC [57]. In the implementation of the algorithm we ran 24 chains in parallel, used a burn-in period of 3000 iterations, and then ran the sampler for 3000 more iterations. Although each chain was individually assessed for convergence, estimates were formed by collapsing across all 24 chains and all 3000 samples, resulting in 72,000 samples of the joint posterior distribution.

To get a sense of whether the model fits the data well, we examined the posterior predictive distribution (PPD) and compared it to the observed data. The PPD is the marginal distribution of new, unobserved data given the data already collected. It gives a prediction about how new data will be distributed if we were to collect more. In a hierarchical model, we can generate the PPD on a subject level or a group level.

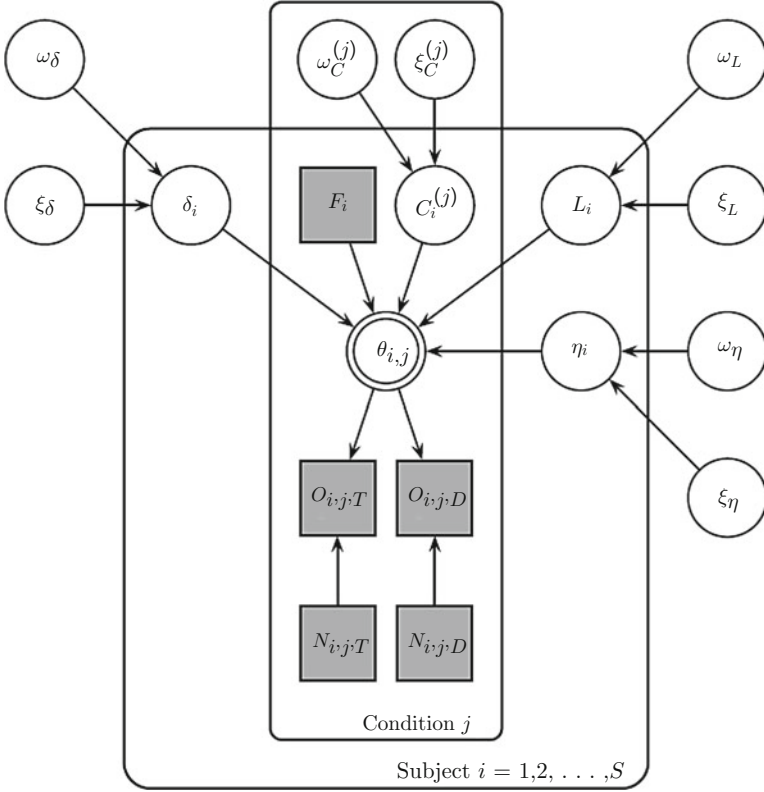


Fig. 3.3 Graphical model for the hierarchical version of the Minvera 2 model fit to the data of [4]. Parameters in the model are represented as white nodes, observed data variables are represented as gray nodes, and deterministic variables are represented as double-bordered nodes. Discrete variables are represented as square nodes, whereas continuous variables are circular. Plates illustrate a replication of a structure within the model, such as parameters across subjects or across conditions

Figure 3.4 shows the PPD at the group level separated across the four conditions. The first column corresponds to conditions in the experiment when no filler was present (i.e., $F0$), whereas the second column corresponds to conditions when the filler was present (i.e., $F1$). The rows correspond to the two list length conditions, where $L0$ denotes the short list condition (i.e., 20 words), and $L1$ denotes the long list condition (i.e., 80 words). In each panel, the black dots correspond to the observed data from Dennis et al. [4] and the gray densities correspond to the PPD. Figure 3.4 shows that the PPD is extremely variable, spreading across the majority of the ROC space. However, this is also true of the observed data [94]. Figure 3.4 assures us that the predictions of the model, which are derived from the fits, are at least sensible in that none of the observed data points fall in a location in the ROC space that is not predicted by the model.

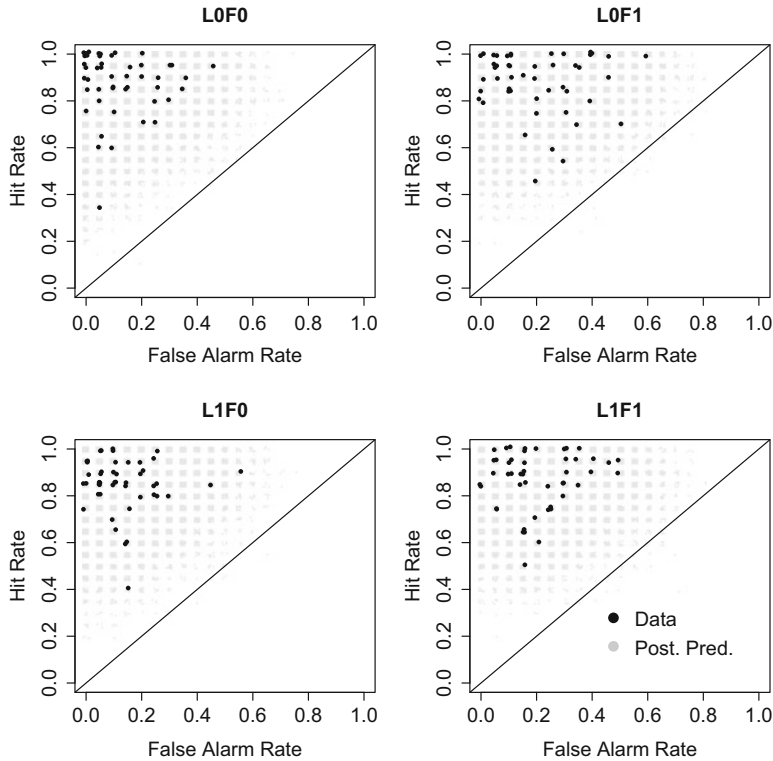


Fig. 3.4 The posterior predictive distributions (PPD) from the hierarchical Minevera 2 model. Each of the panels corresponds to a condition in the experiment from [4]: the columns correspond to the two filler conditions (i.e., filler absent condition $F0$ in the left column, and filler present condition $F1$ in the right column) whereas the rows correspond to the two list length conditions (i.e., the short list condition $L0$ in the top row, and the long list condition $L1$ in the bottom row). Observed data are represented as the black dots, whereas the gray density represents the PPD

Having assured ourselves that the model was fitting the data properly, we examined the posterior distributions. Although there are many parameters we could inspect, we focused on the group level hypermean parameters. Figure 3.5 shows the estimated posterior distributions for the four criterion parameters (top row), the learning rate parameter (bottom left), the decay parameter (bottom middle), and the number of features (bottom right). Figure 3.5 shows that the learning rate parameter for these subjects was quite high, with a mean of 0.955. The decay parameter was very low, with a mean of 0.019. Together these two parameters are important determinants of overall accuracy in the model, and these values in particular help to fit the data (see Fig. 3.4). The number of features parameter is relatively low, having a mean of 8.47. However, from our simulation study above, we now know that when the learning rate parameter is high, fewer numbers of features are required for the model to capture high accuracy data (see Fig. 3.2).

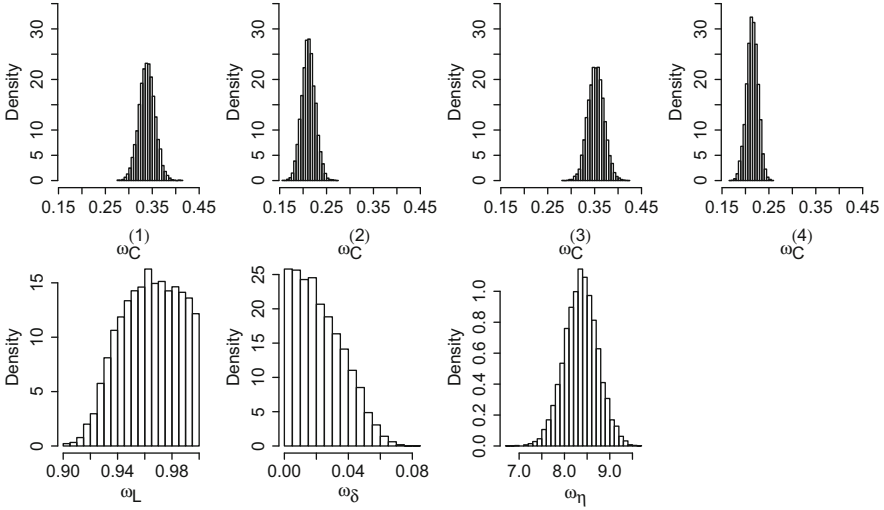


Fig. 3.5 Hypermean parameters for each of the parameters in the hierarchical Minerva 2 model. The top row plots each of the four criterion parameters $\omega_C^{(j)}$, whereas the bottom row plots the posteriors for the learning rate parameter ω_L (left), decay parameter ω_δ (middle), and the number of features parameter ω_η (right)

The two most important parameters to explain the experimental effects are the decay rate δ and the criterion parameter C . Figure 3.6 shows the estimated posterior distributions of the group-level decay rate (left panel) and the difference in the average criterion parameters (right panel). The average criterion difference was computed by collapsing over the criteria for the different list length conditions. Collapsing was justified because Fig. 3.5 shows the posterior means and variances of the decay rates are similar in the first and third conditions (short and long list lengths for no filler task) and in the second and fourth conditions (short and long list lengths for the filler task). If there was no difference in the activations across filler conditions, the estimated posterior of the average criterion difference should be centered at zero. However, Fig. 3.6 shows that the estimated posterior distribution of the difference has a mean of 0.13 and a 99% probability interval that does not contain zero. This suggests that, for the filler conditions, although the decay rate was low, it affected the activations to the extent that the criterion parameter had to shift to maintain the correct pattern of hit and false alarm rates.

We can use the estimated model parameters to gain some insight into the activation distributions used by the subjects. To do this, we generated predictions for the activations in the model for targets and distractors, conditional on the estimates of the group-level parameters (see Fig. 3.5). Figure 3.7 shows the distributions for targets (red) and distractors (gray) when the filler task is present (right panel) and absent (left panel). The criterion parameter is shown as the solid vertical line. Examining the distributions across the conditions provides better insight into why

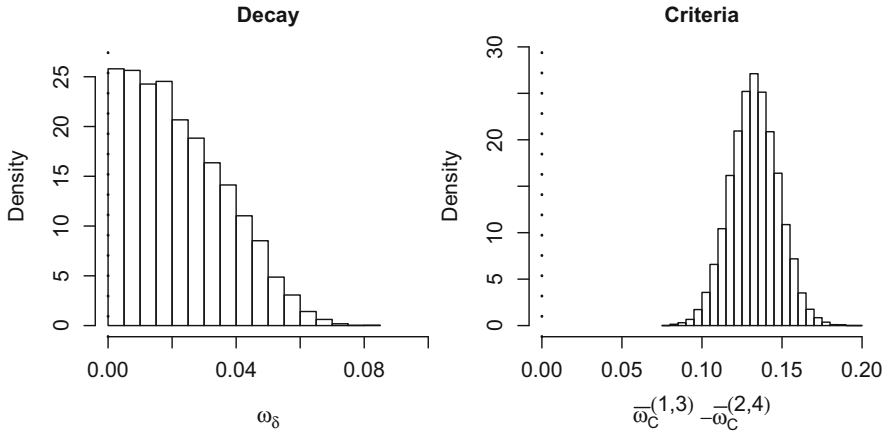


Fig. 3.6 Estimated posterior distributions of the parameters involved in capturing the filler effect. The left panel shows the group-level decay rate parameter, whereas the right panel shows the difference in the average criterion parameters in the two filler conditions. In both panels, the vertical dotted reference lines indicate the value of the parameter that would produce no filler effect

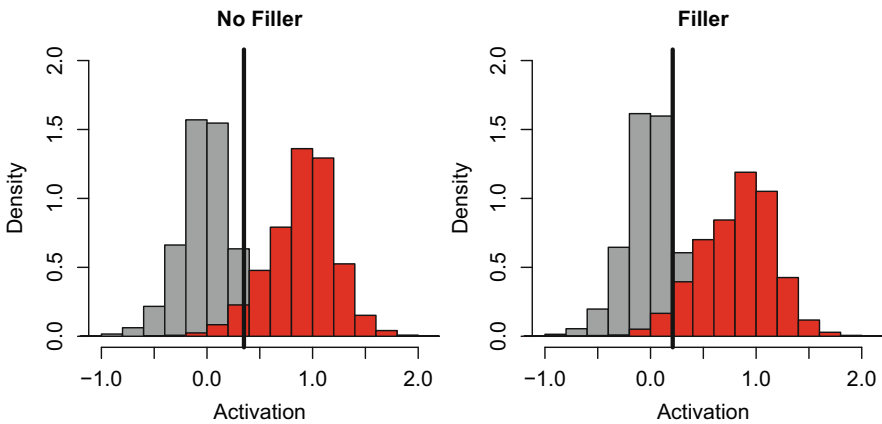


Fig. 3.7 Target and lure activation distributions used by the model across filler conditions. Target activation distributions are illustrated by the red histograms, whereas lure activation distributions are illustrated by the gray histograms. The black vertical line represents the criterion parameter used in each condition

the criterion is lower in the filler condition. Specifically, when the filler task is present, the target representation decays away, and the target activation distribution moves closer to the lure distribution. As a consequence, the criterion parameter needs to adjust to maintain a hit rate that is consistent with the observed data. Hence, together Figs. 3.6 and 3.7 illustrate an important interaction that occurs between the criterion parameters and the decay rate parameter.

3.5 Summary

In this chapter, we illustrated how likelihood-free techniques can be applied to the Minerva 2 model of recognition memory. We began by estimating the joint posterior distribution of the model parameters by generating synthetic data from the model and fitting the model to these data. We compared the estimates of the joint posterior distribution obtained using three different techniques: probability density approximation (PDA) [38], analytic expressions [37], and kernel-based ABC (see Chap. 2). We showed that while the estimates obtained using the two likelihood-free approximations converged to similar values, the analytically convenient approximations diverged from the other methods. Finally, we applied a hierarchical version of Minerva 2 to data from [4] and examined the estimated posterior distributions of the model's hyperparameters. This exercise shows how useful the likelihood-free techniques can be for the Minerva 2 model, a model that has never been incorporated into a hierarchy or fit to data using Bayesian techniques.