Lecture Notes

Lenard Dome

July 15, 2025

Contents

1 Why model?

Let us look at a problem first. Imagine it is Friday night, you are done for the week, and I mean you are wasted. You cannot be bothered to cook, so you order a pizza.

2 The bandit problem

Now, imagine you are a participant in an experiment. Your job is to maximise rewards by choosing between two options, between two slot machines. You choose one, and you get a reward. You choose the other, and you get a reward. You do this for a while, and you start to notice that one of the machines gives you more rewards than the other. This leads you to an understanding of the environment you are in, which means that you learnt something about the rewarding structure of the environment (the mapping between different slot machines and rewards). This is called the bandit problem, and it is a classic problem in reinforcement learning, economics and decision-making. This task is well studied, and it is a very standard task to use when you want to investigate learning processes in humans. We will be using this task in the workshop, and we will use it to illustrate how computational models can be used to understand learning processes.

3 What are we interested in for the bandit problem?

So, we have an experiment, let us figure out what is our theories about what is happening in people's heads. For our current purposes, we will skip the thinking part and go straight to assuming two psychological processes here:

The first is learning. We assume that people learn about the rewards of the two slot machines, and they adjust their behaviour accordingly. We hypothesise that it is an error-driven learning process, such that humans make predictions about what will happen if they choose a certain option, then calculate the difference between their prediction and the reward or feedback they received, and use a portion of this difference to update their belief about the rewardingness of a given option. This is a very common assumption in cognitive psychology, and it is often used to explain how people learn from experience. So, we will use a simplified version of the most famous equation in psychology, the delta rule. This process has its accompanying parameter, the learning rate α, which determines how much you update your belief about the rewardingness of a given option based on the feedback you received. We will look at how high and low learning rates affect the learning process in the first exercise.

The second is decision-making. We assume a simple ratio rule where we treat it as a non-competitive decision-making process, so that options do not compete with each other, but rather the decision is made based on the expected value of each option. This is a again a very common assumption in cognitive psychology, and it is probably the most common way to model decision-making in cognitive psychology. It is going to be a simple SoftMax rule, which is a more sophisticated version of the ratio rule, also called Luce's choice axiom. This rule has its own parameter, the inverse temperature β, which determines how sensitive you are to value differences between choice options.

Now that was quite a bit of jargon, and names, and maybe abbrevations as well. The reason I am mentioning these things here is that so you can go on an google the keywords.

4 Discrepancy functions

Now, we have a model, and we have some empirical findings. What is next? Back in the day, just like in the Rescorla-Wagner example, people plotted or displayed the results of model simulations and empirical data side by side, and then they looked at the graphs and said: "yep, looks good". This works very well if the data you have is robust and very good at distinguishing between models, and if the model is simple, but it is not a very good way to quantify how well the model approximates the data, especially if there are alternatives.

Here come discrepancy functions. Discrepancy functions are mathematical functions that quantify the difference between two sets of datapoints. They are often used to compare the predictions of a model with observed data, and they can help us determine how well a model **fits** the data.

So, we have a problem at our hand. We have the data, Y, the model M, and we want to know how close the two gets.

(next slide)

The first set of discrepancy functions are the ones that are based on the sum of squared differences. These functions are often used in regression analysis, where we want to minimize the difference between the predicted values and the observed values. The most common form is the least squares error, which is simply the sum of the squared differences between the predicted values and the observed values. You can also use them in computational modeling to capture how well the model is doing. These are non-parametric methods, which is great, because they do not make any assumptions about the distribution of the data. However, they are sensitive to outliers, which can skew the results.

But most often, we will use something else, called likelihoods. There are a few reasons for using them. First, models often assign probabilities to the data. On a given trial, in an experiment, there is a certain probability of observaing a certain outcome. Like the probability of picking one from two alternatives (one red pill and one blue pill). The insight that powers likelihoods is that there is a function that can relate this model-assigned probability to the observed data.

For example, if the model predicts that the probability of picking the red pill is 0.7, and the observed data shows that the participant picked the red pill, then the likelihood of this outcome is 0.7. If the participant picked the blue pill, then the likelihood is 0.3 (which is a bad fit). The likelihood function captures this relationship between the model's predictions and the observed data.

Of course, this is all conditional on the model, M, and its current parameters, θ. So, we can write the likelihood function as a function of the model and its parameters, which is what we do here.

Now there is a distinction between two ways these probability functions can be defined. The first, on the left, is when the model actually predicts a probability distribution over the data. This is the case for a Wald Drift Diffusion model, for example. The second, on the right, is when the model predicts a single value for each data point, such as the probability of a given outcome (0.7 chance of taking the red pill). We will then need to define a probability distribution over the data, for example a Bernoulli distribution for binary outcomes (like picking one of two pills).

Today, we will focus on the second type, where the model predicts a single value for each data point, and we will use a Bernoulli distribution to define the likelihood function. This is because it is easier to implement and understand, and it is sufficient for our purposes.

So, I outline here how this is done for a binary outcomes. We have our model assigned probabilities given the moddel parameters, θ, and the data, Y. Using the Bernoulli distribution, we can define the likelihood function as the product of the probabilities of the observed outcomes. This is done by taking the product of the probabilities for each data point.

5 Parameter estimation

So, we have a model, and we have a discrepancy function that quantifies how well the model fits the data. What is next? The next step is to estimate the parameters of the model that best fit the data. We are (for our current purposes) not interested in all possible model output values for all possible parameter values, but rather want to know the parameter values that make the model fit the data best. We are trying to find the peak of the likelihood function, the maximum of the likelihood, given the data and the model. This is called maximum likelihood estimation (MLE).

Most optimisation procedures rely on minimising an objective function, so we will actually minimise the negative log likelihood, which is a common practice and is shown here. There are all sorts of advantages, mainly numerical stability, but it is enough for you now to know that this is what we will do.

So now we have the goal, what is the assumption here? Models that approximate the data well can tell you something about the utility of the model. Essentially, we assume that the model is a good approximation of the data, and that the parameters of the model can be estimated from the data. This is a strong assumption, but it is often necessary to make in order to use computational models in psychology (computational psychiatry). If you are interested in other ways, ask me during the exercises or in the break.

Why do I say that? Because there are caveats to this. Goodness-of-fit does not capture a lot of unexplored aspects of models. Something I have quite a bit of experence with.

Let us start with flexibility. A standard rule of thumb says that the number of free parameters provide an upper bound for model flexibility... if a theory has five orthogonal free parameters, then it will be able to fit exactly any five data points; if the parameters are not orthogonal, however, the number of data points the theory can fit exactly is less. Now we will talk about whether the number of parameters are a good indicator of model complexity - briefly, it is not, and it has been empirically shown multiple times that the number of parameters is often not a good indicator of model flexibility (or complexity) when we talk about cognitive models.

Goodness-of fit also doesn't tell you about how diagnostic your experiment is. Simply put, it requires us to understand what are all the predictions the model can make in a given experiment. Are there things the model does not predict? Can the experimental design allow for those results as well? It also requires us to properly explore variability in the data, and also to show it. Now we usually do this by fitting on a subject-level (fit the model against each subjects data separately).

The last point I want to make here is that goodness-of-fit does not tell you about what is sometimes called as "plausible" falsifiability - are there any results the model could not fit? From a slightly different approach, something I prefer, if the model can account for the data as a function of it being able to account for all possible data, then it is a bad model. A model that is able to produce all logically possible results, observed and unobserved, can never fail to accommodate a result. Such an overly flexible model is inadequate through a lack of specificity.

6 Parameter recovery

After our brief walkthrough from model construction and parameterisation, parameter estimation, and discrepency functions, I would like to talk a bit about a way to evaluate how reliable your model is before any data has been collected. This is called parameter recovery.

Mathematical models often reflect real-world systems and their parameters have biological, chemical, or physical interpretations, and not identifying these parameters can result in ambiguous interpretations. Parameter recovery is a method to assess the reliability of parameter estimates by simulating data from a model with known parameters and then attempting to recover those parameters from the simulated data.

This method comes in handy when you need to determine whether your model can accurately estimate the parameters of interest - you have to determine whether model parameters are identifiable, meaning that a given data can be only capture by unique parameter values.

Parameters are identifiable when there is only one possible value given the data.

Parameters are not identifiable, when there are multiple possible values that can explain the data. This can happen when the model is too flexible, or when the data is not informative enough to distinguish between different parameter values. As models become more complex, like some of the more complex recurrent networks and cluster-based categorisation modesl I dealt with, evaluating this becomes difficult, because the same dataset can be produced by multiple parameter sets, each corresponding to different underlying representations in the model - even though the model can produce the same data with different parameter sets, its internal state can differ, which corresponds to different predictions about what underlies a certain behavioural response. For the model you will use in the workshop, this is not a problem, because the model is simple enough to not have to worry about it.

This is an overly simplistic view of what parameter recovery and indetifiability means... the more time you spend with models, the more you will realise that it is not always that simple. In a Bayesian framework, or even with simple parameter estimaton, there is uncertainty involved in the estimation process.

6.1 Ben. J. Wagner on parameter recovery

Even with maximum likelihood estimate, which is your best guess, but this estimate is always uncertain. In a Bayesian framework only -one- param value is at odds with Bayesian inference where the output is a distribution. Likewise a well-identified parameter in almost all cases has multiple possible values, which are more or less probable. So I think its better to say that the problem with a non-identifiable parameter is when the data does not provide info on how to distinguish between those multiple suggestions/our beliefs dont become more precise.

I would therefore suggest something more specific (because its an important topic):

Parameter identifiability: can we learn from the data? Identifiability addresses whether a parameter's value can be effectively estimated from the observed data. In Bayesian modeling, this is about reducing uncertainty about the parameters true value.

Identifiable parameter: The data informs our beliefs. The posterior distribution is narrower and more constrained than our initial prior distribution or with MLE: has a likelihood function with a single, well-defined peak. This means there is a parameter value that maximizes the likelihood of observing the data. As you collect more data, the peak should become sharper and more distinct

Non-identifiable parameter: The data provides little to no information. The posterior distribution looks very similar to the prior, meaning we haven't learned much about the parameter from the data. Or with MLE: results in a likelihood surface that is rather flat. This means different parameter values (or combinations of them) produce the exact same MLE, making it impossible to choose a unique best-fitting parameter. You can't distinguish between them based on the data.

Further on recovery: A parameter must be theoretically identifiable so that we can have any hope of it also being recovered. But identifiability alone doesn't guarantee successful recovery. That also depends on the model and correlation between parameters. e.g. can different combinations of parameters explain the data as well? Model complexity and so on.

6.2 Return to main narrative

But for our purposes, this is a good starting point.

This here shows the result of a prototypical parameter recovery procedure. We have two parameters, learning rate and choice stochasticity - don't worry too much about what they mean, you will be explained in the worksheets.

The Pearson correlation is quite high, which means that parameters are identifiable.

7 Model comparisons

So far, we only looked at ways that quantify how well the model does by itself. Now we shift towards looking at how well the model does compared to other models. Model comparison is crucial, especially because we often want to exclude potential theories in order to arrive at an adequate model of whatever we are trying to model.

The first issue I want to talk about is **overfitting**. You have probably heard this word many times and it is often used in the context of machine learning, but it is also relevant for computational modeling. Overfitting occurs when a model fits the training data really well but may perform poorly for independent data. Sometimes it is due to model complexity, but it largely depends on how you view complexity or how you measure it. It can also happen due to overtraining models, which is more of a problem in larger neural networks. But the problem is that the model does not generalize well to new data, and it is therefore not a good representation of the underlying system.

On the right here, this is demonstrated really well. You have three models with increasing complexity (increasing number of parameters) trying to predict life satisfaction scores as a function of years in marriage. You can see that the more complex models fit the training data better, but they do not generalize well to new data - no one expects satisfaction with life to drop to 0 after 10 years of marriage, but the most complex model predicts that. So, what does this mean for us? If we stick with the complexity dimension, we hav e a push towards simpler models, because they are less likely to overfit the data. But we also have a push towards less model parameters. Usually in the case of less than the number of data points.

This is the way we usually approach the problem of overfitting - through complexity as measured by the number of parameters. Unfortunately, the flexibility added by a free parameter depends on the details of the theory (cf. $\alpha x + \beta x$ with $\alpha x + \beta$; both have two parameters, but the latter is more flexible). The only accurate way to "allow" for the flexibility of a theory, as far as we know, is to determine what the theory predicts, and for that you have to simulate. Some methods try to address this shortcoming by looking at incorporating variance explained by each parameter into the penalty term. These include information criterion measures that generalize the goodness-of-fit measures across the model's parameter space, assuming that the data is large enough and the parameters are distributed according to Jeffrey's prior. And other measures that try to add measures on how well each parameter

Another important aspect of model comparison is distinguishability. The problem we are facing here is that we often have multiple models and want to be able to identify the one that is better at explaining the data. Model recovery is a method to assess whether we can identify which model generated the data. This is done by simulating data from each model and then fitting the models to the simulated data. Then we can count how many times each model came out on top when fitted against the simulated data, which gives an indication of how well the models can be distinguished from each other. We will talk about it in a second in more details.

Some extra techniques I want to mention because I think it is important to be aware of them, even though we will not cover them, are landscaping and parameter space partitioning. Landscaping is a way to extend model recovery by comparing model fits against each other visually. It is a way to check how well one model is preferred over the other in a given comparison. Parameter space partitioning is a little bit more sophisticated. In this approach, we turn model predictions into a finite set of countable discrete outcomes, and we use those to identify regions in the parameter space that correspond to different discrete outcomes (such as different response patterns in a categorization task). This method gives an interesting measure of complexity, that is based on actual model outcomes as opposed to the number of parameters.

8 Model recovery

We will talk about model recovery in more detail, because this is good and standard practice that most people should do before data collection or when planning to develop a research programme or a series of studies that involve model comparison.

In a model recovery, we extend what we learnt about parameter identifiability to models. In this scenario, we generate data from a model with known parameters, and then we fit multiple models to the data. We repeat this process for all data generating model many times, and estimate how likely that the model wins over the alternatives for a given dataset. So, we acquire what is often called confusion matrix, see on the right. Each cell gives you a probability of a model given the data, and the rows are the data generating models. The diagonal gives you the probability of the true data-generating model winning over the alternatives, and the off-diagonal cells give you the probability of the model is losing to the alternatives.

If the models are distinguishable, then the diagonal cells will be high, and the off-diagonal cells will be low. If the models are not distinguishable, then the diagonal cells will be low, and the off-diagonal cells will be high or comperable. Now it is often the case that models are misspecified and become indistinguishable, like when they share the same parameters or parts of their architecture. Or it could also be the case that the experimental design is not complex enough to distinguish between the models. In this case, we have to return to the drawing board and either change the model or the experimental design.

Now beware that what we mean by confusion matrix here is not what we mean by confusion matrix in machine learning. It is an important distinction, but they can be related under certain conditions.