Introduction to Bayesian Generative Modeling

Marco Wirthlin

May 2019

In this section, I will outline how Bayesian computational modeling of a wide range of phenomena can be approached, starting with a simple example and building up the modeling procedure.

1 A basic example

1.1 Implementing a hypothesis

Models can be conceptualized as hypothesis regarding how a set of *measurements* (data from experiments or any other source) relate to *observed phenomena*, in real world settings. In other words, a model offers an explanation of *why* a certain set of measurements were observed. Models try to address situations where the data is known, but not the processes that generated it nor its properites. Such circumstances are also called the *inverse problem*. In mathematical models, those hypothesis are formulated as mathematical expressions. As an example, adapted from Shiffrin et al. (2008), one could hypothesize that the *retention rate* ($\mathtt{RR}$) of learned word lists has a certain decay. This is a verbal hypothesis and does not stipulate the *dynamics* of the decay or assign any quantities to it. In order to learn how fast the retention of the words fades, different groups of participants could be asked to recall a word list at different intervals after a learning phase. Word $\mathtt{RR}$ should be lower, the longer the interval between recall and learning ($\mathtt{TI}$). While this hypothesis predicts the outcome of the experiment in general terms, it cannot describe a deterministic system of relationships between $\mathtt{TI}$ and $\mathtt{RR}$. A third, mechanistic, element is required to be added to the hypothesis which links dependent and independent variables. In this example – not minding any domain knowledge from memory research – it is at hand to characterize any observed decrease in $\mathtt{RR}$ as an exponential decay, such that:

$$\mathtt{RR} = \mathtt{TI}^{-d} - a \tag{1}$$

where d is the decay or ”forgetting” parameter and a stands for the initial forgetting. This formulation is a mechanistic, theoretical aid which replaces verbal hypothesises for its specificity and ease to contrast with observed data. Because such definitions usually are developed inside a considerable corpus of scientific knowledge, it is important to formulate them in concordance with the literature. One way of achieving this is by relating the parameters with other variables of interest. In this example, incorporating a variable such as *word frequency* might be beneficial as it impacts $\mathtt{RR}$ directly. Another way of improving the current formulation might might be by adding details of *processes*: For example, the decay d might not be stationary, but increase linearly as time passes. This process of building and refining principled mechanistic and mathematically explicit, explanatory mechanisms of an observed phenomenon, enriched by domain specific knowledge, can be called (mathematical) modeling Melnik (2015), Gelman et al. (2013). Note here that the goal is not to create generic statistical models in the traditional data analysis sense, such as the generalized linear model for statistical inference, but relating the model to data. The difference lies in that data analysis models usually are applied for relating variables (the measurements) amongst each other and/or for inferring population parameters, while models that aim to provide a mechanistic explanation of cognition are used to infer latent (not observed) parameters (Lee, 2011).

By constructing mechanistic relationships between latent parameters and observed phenomena, the nature of such models is inherently *generative* (Love et al., 2015). The aims of cognitive science and psychology is to build models which can generalize to every human, in as many situations as possible. For business, such models can be used to predict as many outcomes, as accurately as possible while understanding how such predictions came to be. *Discriminative* solutions, while performing better in the long run (specially when more data is available) (Ng and Jordan, 2002), are not suitable for solving the inverse problem and creating predictions when a big amount of data is difficult to access, gather, computing infrastructure is not available or the data is otherwise expensive. This is especially the case in clinical studies or when neurophysiological measures are involved and subjects' movement are restricted. In addition, models and algorithms which produce direct mappings ($f : \mathcal{D} \to \hat{y}$, or equivalently $p(\mathcal{D}, \hat{y})$) such as logistic regression or connectionist networks (neural networks) provide little *explanatory power* regarding how a phenomenon is related to observed variables. By contrast, in generative models, all parts of the model (data, measurement process, likelihood and posterior) need to be modeled explicitly.

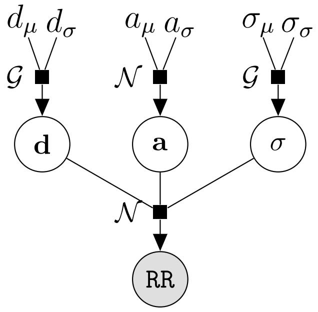

Figure 1: *A directed, acyclic graph, representing the probabilistic model from Equation 2.*

1.2 Bayesian inference procedure

An approach adopted in most generative models – the Bayesian approach – is to quantify all model components, specifically their uncertainties, with probabilities. In such models, it is possible to estimate a probability distribution for each parameter, *conditioned* on the data, called the *posterior distribution* and denoted $p(\theta|\mathcal{D})$. Reformulating the previous example from Equation 1 probabilistically is the first step in such a procedure and will result in:

$$\begin{aligned} d &\sim \mathcal{G}(0.1, 0.01) \\ a &\sim \mathcal{N}(0.2, 0.1) \\ \sigma^2 &\sim \mathcal{G}(0.1, 0.01) \\ \texttt{RR} &\sim \mathcal{N}(\texttt{TI}^{-d} - a,\, \sigma^2) \end{aligned} \tag{2}$$

$\mathcal{N}$ means a Gaussian and $\mathcal{G}$ Gamma distribution respectively. In equation 2, as each part of the model is expressed in probabilistic form , the assumptions regarding those quantities are made explicit: The retention rate of words ($\texttt{RR}$) is thought to be distributed normally among the subjects (denominated by $\mathcal{N}$), with the mean being calculated as in equation 1 and a variation of σ^2. What makes this model a Bayesian one is the inclusion of the priors for d, a and σ^2. Those quantities are motivated by theory and literature. Priors introduce an (additional) subjective facet into model building which allow for specification of externally imposed "real world" constraints concerning observed phenomena. The more data is available, the less their influence weights in into the final result. Assumptions are made explicit in form of the probability distribution: d is expected to be distributed gamma-shaped, while a and σ^2 follow a normal distribution (here the quantities are chosen arbitrarily, but should be informed by domain knowledge in a real setting). Priors are denoted more generally as $p(\theta)$. Note that θ stands for all parameters (Griffiths et al., 2008).

The last expression in Equation 2: $\texttt{RR} \sim \mathcal{N}(\texttt{TI}^{-d} - a,\, \sigma^2)$ is often called *likelihood*, and is denoted more generally as $p(\mathcal{D}|\theta)$. It stands for the probability of the observed data $\mathcal{D}$ ($\texttt{RR}$, as observed in the experiment), *conditioned* by certain parameter values. In other words, the likelihood is a function that returns the probability that a certain datum pertains to a distribution, described by the parameter values: For instance, the probability of the value 9 to be observed under a distribution described by $\mathcal{N}(\mu = 3, \sigma = 0.1)$ is 0, while it will increase, the closer μ (the mean of the distribution) gets to 9. In likelihood functions, the datum is constant while what is varied are the parameter values. To compute the overall likelihood for a *certain combination of parameter values, for multiple observations* ($\mathcal{L}(\theta)$), the likelihoods for each individual datum have to be multiplied with each another. As large multiplications are computationally expensive and potentially unstable, the likelihood for a set of parameters can be computed by taking the natural logarithm (resulting in the *log-likelihood*) of each instance and making the overall sum:

$$\ln(\mathcal{L}(\theta)) = \sum_{n=1}^{N} \ln p(\mathcal{D}|\theta) \tag{3}$$

for i = 1,...,N, which is the amount of observed data points. In general, even after $\ln(\mathcal{L}(\theta))$ has been computed, $p(\mathcal{D}|\theta)$ is used to denote the overall likelihood. If the parameters are orthogonal, then $p(\sigma)p(d)p(a) = p(\sigma, d, a)$

With priors and likelihoods, it is possible to calculate the probability that both data and parameter are matched (the probability that both instances are co-occurring). This is called the *joint probability distribution* $p(\mathcal{D}, \theta)$, and is calculated by multiplying the likelihood by the priors: $p(\mathcal{D}|\theta)p(\theta)$. The joint probability for the probabilistic model from Equation 2 is following:

$$p(\mathcal{D}, \sigma, a, d) = p(\mathcal{D}|\sigma, a, d)p(\sigma)p(d)p(a) \tag{4}$$

In order to calculate the posterior distribution, the joint probabilities have to be normalized with the *marginal likelihood* or *evidence*, which is the average for each datum, of the likelihood for every value of each parameter, weighted by the prior probabilities (Kruschke, 2014). The general form for the marginal likelihood when parameters are discrete is the sum over all parameter instances, but most cases, parameter are continuous: $p(\mathcal{D}) = \int \mathrm{d}\theta_k p(\mathcal{D}|\theta_k)p(\theta_k)$ for k = 1,...,K, which is the range of parameter values for each parameter. It is called the "marginal" likelihood as the contributions from the parameters and priors have been "integrated out" and is reduced to the probability of the data, under the current model (Kruschke, 2014). The marginal likelihood $p(\mathcal{D})$ for the probabilistic model from Equation 2 is following:

$$\iiint p(\mathcal{D}|\sigma_k, d_k, a_k)p(\sigma)p(d)p(a)\mathrm{d}\sigma_k\mathrm{d}d_k\mathrm{d}a_k \tag{5}$$

With priors, likelihood and evidence, it is possible to calculate the posterior distribution $p(\theta|\mathcal{D})$ via *inverse probability*, also known as Bayes' theorem (Gelman et al., 2013). Is represents the ratio of probability that data and parameters co-occur, and the probability that the data is present under the current model. By having the evidence in the denominator, the joint probability gets conditioned by the datas' probability, thus, re-allocating probability (credibility) from a data-agnostic distribution to one that includes the data measurements. Its general form is $p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})}$. Applied to our example from Equation 2, the posterior distribution $p(\sigma, a, d|\mathcal{D})$ for each parameter is calculated by combining the elements from Equation 3, 4 and 5:

$$\frac{\overbrace{\sum_{1\ldots N} \ln(p(\mathcal{D}|\sigma_k, d_k, a_k)}^{Likelihood}\overbrace{p(\sigma)p(d)p(a)}^{Priors}}{\underbrace{\iiint p(\mathcal{D}|\sigma_k, d_k, a_k)p(\sigma)p(d)p(a)\mathrm{d}\sigma_k\mathrm{d}d_k\mathrm{d}a_k}_{Evidence}} \tag{6}$$

Estimating parameters' posterior distributions has the advantage over single point estimates – such as maximum likelihood estimation, used in traditional frequentist approaches – that the distributions allow to determine how (un)certain the estimates are. This information is valuable when evaluating the models performance in terms of specificity: Should all parameter values be similarly probable under a wide, light-tailed distribution, the model is not providing helpful information. Similarly, if the posterior resembles the prior distribution, the contribution of the data in terms of credibility re-allocation is too weak. In both cases, data quantity and quality, and the choice of the priors has to be minded. Extremely wide "non-informative", but also too strongly informed priors with a very low variance might overweight the information of the data. Another option is that, even if certain prior yields apparently sufficient posterior distributions, it might not be in line with the domain knowledge the model is rooted in. The correct choice of the prior is a complex issue which is still a topic of ongoing discussion in the literature Gabry et al. (2019). In (Gelman et al., 2017), this issue is discussed in detail.

1.3 Bayesian Networks for model representation

The probabilistic model from Equation 2, serving as example for the bayesian model construction procedure followed in the present work is quite simple. Many models of cognition, economics, engineering or physics are more complex in terms of number of parameter and mechanistic explanations of phenomena. As the number of variables and parameters grow, computing and understanding the joint probability distributions for those probabilistic models becomes unpractical (Griffiths et al., 2008), Pearl (2009). Bayesian Networks (BNs) aid in handling high-dimensional probability distributions. They are a specific form of directed acyclical graphs (DAGs) and allow to represent the dependency structure and interactions

$$\begin{aligned}&\text{Hierarchical extension: } \prod_{i}^{N} \overbrace{p(\mathcal{D}|\sigma_i,a_i,d_i)}^{\textit{Likelihood}}\underbrace{p(\sigma_i)p(a_i)}_{\textit{Priors}}\overbrace{p(d_i|d_\mu)p(d_i|d_\sigma)p(d_\mu)p(d_\sigma)}^{\textit{Hyperpriors}}\\&\text{Multilevel extension: } \prod_{i}^{N}\prod_{j}^{M} p(\mathcal{D}|\sigma_i,a_i,d_i)p(\sigma_i)p(a_i)p(d_{\mathbf{ij}}|d_{\mu_{\mathbf{j}}})p(d_{\mathbf{ij}}|d_{\sigma_{\mathbf{j}}})p(d_{\mu_{\mathbf{j}}})p(d_{\sigma_{\mathbf{j}}})\end{aligned} \tag{7}$$

between assumptions, probability distributions and observations. With BNs, it is possible to represent all the assumptions regarding the data generating process in a graphical manner (Pearl, 2009).

Figure 1 depicts a graphical representation of the probabilistic model from Equation 2. Black squares represent the distribution types, which are denoted by the letters to their left. Deterministic quantities are denoted as variables. They are constants which configure the priors. In figure 1, d_μ and d_σ are inputs for a gamma distribution, which gives rise to d. The same logic can be applied to parameters a and $sigma$ and is the graphical equivalent of stating: $d \sim \mathcal{G}(d_\mu, d_\sigma)$, where $d_m u = 0.1$ and $d_\sigma = 0.01$. White circles contain probabilistic quantities, given rise to by parent distributions, as explained above. Shaded circles represent observed data. Thus, the distribution node (black squares) connecting to the data node determines the form of the likelihood. Note that the black squares, also called "factors" are a recent addition to DAGs and BDs by (Dietz, 2010) and offer easier interpretation of the model by incorporating which distributions mediate between parent and child nodes. Factors are merely a visual aid and do not affect the inference procedure in any way.

BNs are directed, which means that they represented probabilistic relationships in a unidirectional fashion via their edges. Furthermore, BNs are acyclical: None of the relationships feed back into a previous node, so that there is no recurrence. What distinguishes BNs from DAGs are two sufficient and required *Markov conditions*. First, parent nodes have to condition child nodes' variables in such a way that they are independent of all other variables, but further child nodes' variables. The unidirectional edges in BN's which range from priors to likelihoods are consistent with this condition (Griffiths et al., 2008), and as such, all parent nodes are *markonian parents* for their children. The second condition states that the probability functions part of the model need to admit canonical factorization of the complete joint probability distribution into conditional probability distributions, defined locally by parent-child relationships in order to be *markov compatible* (Pearl, 2009). Those conditions are not only in line with bayesian inference, but provide a "road map" for how the parameters' probability distributions interact with each another, depending on their edge-relationships. If those conditions are not met, DAGs are not able to describe a stochastic process capable of generating the data.

1.4 Hierarchical Extension

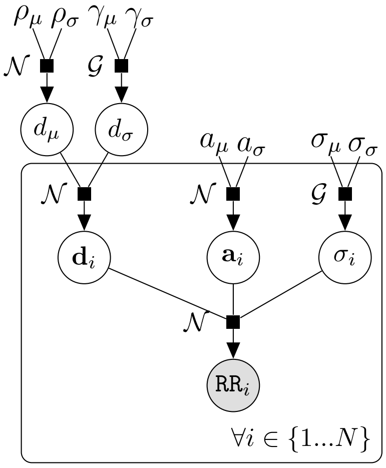

Figure 2: *A hierarchical extension of the probabilistic model from Equation 2.*

The general form of markov compatiability resembles Equation 4 for the joint probability $p(\mathcal{D}, \theta)$: $p(x_1, ..., x_n) = \prod_i p(x_i|pa_i)$, for i pairs of local factorizations between parent-children pairs. This means, that if there is a parameter ϕ, that conditions another parameter θ before multiplying the likelihood, the formulation for the joint probability distribution will be $\sum_{n=1}^{N} \ln[p(\mathcal{D}|\theta)]p(\theta|\phi)p(\phi)$ (Betancourt and Girolami, 2015). Because the model from Equation 2 does not feature any nodes conditioning the probability distribution of the parameters d, a and σ, Figure 2 features a extension to demonstrate how higher order nodes condition subsequent nodes, such as is the case in the models featured later. The addition of hyperparameters and their priors have to be taken into account in the calculation of the joint probability distribution. But, as already mentioned, Figure 2 offers an easy "road map" for the new formulation for $p(\mathcal{D}, \theta)$. The joint probability $p(\mathcal{D}, \sigma, a, d, d_\mu, d_\sigma)$ is calculated as in Equation 7, top portion (Gelman and Hill, 2006), (Pearl, 2009).

Parent nodes which condition parameters can be called *hyperparameters* (d_μ and d_σ in Figure 2). Such nodes have to be set up with their own priors, in this case ρ_μ, ρ_σ, γ_μ and γ_σ. Hyperparameters are also called *hyperpriors* when put in relationship to the likelihood. Hyperparameters introduce a new layer of complexity in model conception: They can stand for mechanisms that give rise to the parameters (processes such as those require additional hypothesises, like in Equation 1), but also be of more statistical nature, as in our memory decay example model from Figure 2: d_μ and d_σ further "split up" the memory decay parameter d into a mean and variance of a normal distribution. In addition, the rounded rectangle surrounding all nodes but d_μ and d_σ symbolizes that those parameters have to be estimated $\forall i \in \{1...N\}$, which means "for each sub-index i, in the range from 1 to N". In this example, N stands for the total number of subjects that participated in the study and i for each individual "subject number".

This model differs from the one in Figure 1, in that instead of having one parameter estimate for all subjects, there will be one estimate for each *except* d_μ and d_σ. Those two parameters, as shown in 7 (top) and Figure 2, have no sub-indices. d_μ acts as a Gaussian mean for *all* subjects, and the same applies to the variance d_σ. Because d_μ and d_σ are not estimated for each subject, their estimates are going to be "pooled" together, "blurring" individual-level differences, treating the data of each subject as they would come from the same one. All other parameters, being estimated for each subject, are "unpooled". Completely pooled models feature less variance, but are more biased because they disregard individual-level idiosyncrasies. Completely "unpooled" models feature very little bias, as each subject has it's own parameter estimate, but increases the variance of the estimates considerably as there is less data available. Data from one subject is not able to inform the parameter estimation of another subject. This can result in estimating more parameters than data. Because this models combines "pooled" and "unpooled" estimates, it is denominated a "partially pooled model" and represents a trade-off between high-bias/low-variance and low-bias/high-variance approaches (Betancourt, 2016).

The hierarchical structure makes ad-hoc calculations on individual parameter d estimates, to obtain its mean and variance, unnecessary. Furthermore, the structure naturally corresponds to the "nesting" of the parameters into the subjects, while d_μ and d_σ describe more abstract group distributions (Shiffrin et al., 2008). Katahira (2016) has determined via simulation studies that hierarchical models deliver more accurate parameter estimates, specially when there is a lack of data for one or several subjects, but also at a greater computational cost when comparing to non-hierarchical parameterizations.

In hierarchical models, lower-level parameters get "pulled" towards the modes of the higher-level distributions, resulting in a reduction of variance among the estimates compared to the data points. This behaviour is called "shrinkage". Shrinkage is stronger for subjects (or any instances of i) which provide less data (Gelman et al., 2013). While it can prevent overfitting by clustering individual estimates around their groups estimate mode, should higher-level distributions feature multiple modes, they also might "pull apart" the estimates and thus increase the variance compared to the data variance. Careful inspection of mixture distributions might help to spot such distributions. Shrinkage, while being a property of hierarchical models, are cause by their structure, not by Bayesian inference procedure (Kruschke, 2014). Shrinkage can also be conceptualized as "regularization" in terms of more traditional machine learning literature.

Another drawback of hierarchical models, besides increased computational requirements, is that relatively small changes in the top-level parameters/hyperpriors produce strong variations in the curvature of the joint probability distribution (Betancourt and Girolami, 2015). This affects mainly the performance of Monte Carlo Markov Chain based algorithms exploring this distribution and will be discussed in the section regarding Hamiltonian Monte Carlo.

1.5 Multi-level Extension

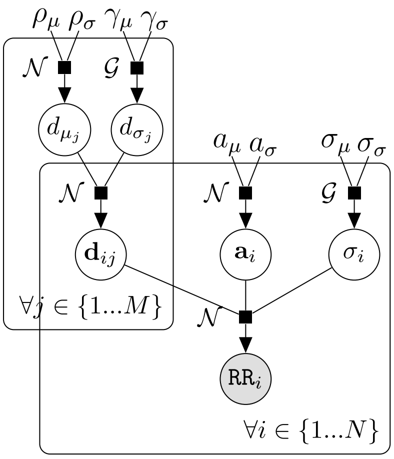

Figure 3: *A multi-level extension of the probabilistic model from Equation 2.*

Multi-level model feature more than one hierarchical structure where parameters are nested (Betancourt and Girolami, 2015). Figure 3 is an example where an additional level j has been added for hyperparameters d_μ and d_σ, but also for the "decay" parameter d. Recall that in the memory decay example, subjects have been asked repeatedly to remember the word list, learned previously. The additional level j is going to vary $\forall j \in \{1...M\}$, where M stands for those measurement times of the word recall rate. This new level demonstrates how parameters can be nested for aspects related to the experimental design, such as repeated measurements or different levels of a certain stimuli. This model will estimate a decay rate for each subject, for each time the subject has been tested. Note that the subject-level estimates and their variance will be "shrinked" together by each measurement moments' estimate. A hierarchical structure such as this comes with the hypothesis that the measurement time is more relevant than the individual level estimate of the decay d. The formulation for the joint density distribution is depicted in Equation 7, bottom part.

2 Approximation of the posterior probability distribution

One of the biggest challenges of Bayesian inference is the calculation of the integral belonging to the *evidence*, as in our example, in Equation 5. For each parameter, the joint probability has to be *marginalized over* the complete distributions. In other words, the computation of the average over all parameter settings of the likelihood, weighted by the (conditional) priors, for a certain data set, also called *expectation* (Kruschke, 2014) and Griffiths et al. (2008). This operation is only tractable (solvable via symbolic manipulation / analytically solvable) should the prior distribution(s) be *conjugate* with the likelihood. When a likelihood is multiplied by a conjugate prior, the resulting function will retain the likelihoods' form. There are only few options available for the most used likelihoods. The Bernoulli likelihood, for instance, only has the beta distribution as conjugate prior, while for the normal distribution, a conjugate prior, is another normal distribution. This limitation restricts model complexity, as only the most simple models, with no mixtures such as in Equations 4 and 7, will yield tractable posteriors. Even if a simple model were sufficient for accounting for a certain phenomenon and corresponding data set, it might not be possible representing prior knowledge by the conjugate priors. Imagine attempting to simulate the exponential growth rate of a bacteria and then distributing the result via a normal distribution. While this would lead to an intractable joint distribution, any other prior, incongruent with scientific literature, is not acceptable. Because of the challenging integral, the general form of Bayes theorem is often expressed as a proportion, leaving out the denominator completely:

$$\begin{aligned} p(\theta|\mathcal{D}) &\propto p(\mathcal{D},\theta) \\ &\propto p(\mathcal{D}|\theta)p(\theta) \end{aligned} \tag{8}$$

As the *evidence* is calculated via the integral over the complete range of all parameters weighted by the priors, (like in Equation 5) and as such, is a average over the parameter space, it can be considered the *expected value* over said parameter space. A solution to when conjugacy is not an option, or the studied phenomena does not allow for simple models, is to approximate the posterior distribution by repeatedly sampling from the joint distribution. In short: the joint distribution $p(\mathcal{D},\theta)$ can be considered the *expectation function* $f(\theta)$ of the probability distribution $p(\theta)$. The integral of $f(\theta)p(\theta)$ can be approximated as following (Betancourt, 2017):

$$\begin{aligned} \mathbb{E}_p[f] &= \int f(\theta)p(\theta)\mathrm{d}\theta \\ &\approx \frac{\sum_i^N f(\theta_i)}{N} \end{aligned} \tag{9}$$

for $1,..,N$ samples from $f(\theta)$. The bigger N, the more accurate the approximation of the posterior distribution (Kruschke, 2014). Evaluating the expectation function $\mathbb{E}_p[f]$ for the target function $p(\sigma)$ (which stands for $p(\mathcal{D})$ in this general formulation) for each combination of parameter values via grid search, while theoretically possible, is impractical in reality. The combination of parameters – the regions of the parameter space to explore – which will result in a good estimate of the expectation and therefore, of the posterior distribution, is where most *probability mass* is concentrated. In a one-dimensional example, this region is usually clustered around the median. But as more parameters are added, the volume far away the multi-dimensional medians "pulls" probability mass away, towards the tails of the distributions, resulting in a "corridor" or "strip" like shape. Most probability mass can be found between the regions which have low-density/high-volume (region far away medians) and high-density/low-volume (regions close to medians). This space is called the *typical set*. When exploring the typical set, the contribution to the expectation is highest. The bigger the parameter space, the bigger become the regions where either volume or density is high and the more the typical set is narrowed down to a "corridor" in parameter space (Betancourt, 2017). Finding algorithms that are able to detect and exploit the typical set is paramount to avoid wasting computational power and time on low-interest regions.

3 Hamiltonian Monte Carlo and the No-U-Turn-Sampler for efficient posterior exploration

Lets define the posterior probability distribution as our *target distribution* $p(\theta)$, approximated by sampling from the joint probability distribution as our *expectation function*. One approach to efficient target distribution exploration is based on *markov chains*. A markov chain is a set of points in the parameter space, where each point depends only on the state (position) of the previous point. Note that this is not related with *markov compatibility* as discussed for the Bayesian networks in this context. Initialization of the chain is usually set on a random location. The *markov transition*, also more generally called *transition kernel*, defines the exploration behaviour of the chain. Should this transition be completely stochastic (at random), the chain will perform a *random walk* through parameter space where each "stop" is called a *sample*. Eventually, after a large number of samples, when averaging like in Equation 9, $p(\theta)$ will be approximated.

To make markov chains more efficient, the transition kernel has to be guided towards the typical set of the parameter space. This is the approach of the Metropolis-Hastings algorithm (Betancourt, 2017). This algorithm also initializes at a random or user-defined, non-zero point in the parameter space, $p(\theta_{current})$. Then, a new position $p(\theta_{proposed})$ is generated by the *proposal distribution*, which is usually a Gaussian distribution with mean 0 and some standard deviation, mostly in the same or similar magnitude than the parameter values. The standard deviation is the "step size" of the algorithm and controls how "far" each iteration the chain will "jump" in the parameter space. Should it be too small, the algorithm might spending many iterations in one small region of the parameter space, not exploring distal regions. If the step size is too big, the algorithm might miss many details. Finding and setting the correct step size manually is a question of trial and error with the Metropolis-Hastings' algorithm and a few tries might be necessary until finding one that works best for a specific parameter space and data set. The acceptance of jumping to the new position is regulated via the following formulation:

$$p_{move} = min\left(1, \frac{p(\theta_{proposed})}{p(\theta_{current})}\right) \tag{10}$$

Should the proposed position feature a higher or equal density than the current position, $p_{move} = 1$ and the proposal step is accepted automatically. Otherwise, p_{move} will be smaller than one. In this case, the proposed position can still be accepted. For this to happen, a random number drawn from a uniform distribution between 0 and 1 has to fall in between the interval $[0, p_{move}]$. This stochastic acceptance/rejection scheme is able to produce samples which approximate the posterior distribution much faster than a random walk would.

Metropolis-Hastings' proposal distribution will work fine for well-behaved, smooth posterior distributions. "Fine" here means that the markov chain explores the typical set for all parameters, spanning the complete "topography" of the posterior.

But the Gaussian proposal distribution in high-dimensional geometries with narrow ”corridors” or ”ridges” will ”overshoot” and miss those concentrations of probability mass many times, leading to a high number of iterations with rejections. Only increasing the number of iterations significantly solves this problem, which leads to increased computing time and does not guarantee good results: The markov chain might either never converge, or, it might get stuck in a small ”ridge”, where it spends all its iterations.

Hamiltonian Monte-Carlo (HMC) is a alternative algorithm with a transition kernel that is ”aware” of the posterior’s topography. The two-dimensional, Gaussian proposal distribution from the Metropolis-Hastings algorithm is replaced by a *Hamiltonian system*. In this setting, the parameter space gets first inverted: now the highest probability mass is situated at the *lowest* point in the parameter space, while uninteresting regions lie high. This way, in high-dimensional parameter spaces, the typical sets form ”valleys” or ”canyons”. In HMC, a particle is simulated at each point where the markov chain lands in the parameter space. The particles’ path is determined by two forces: ”gravity” and the particles ”momentum”. Gravity is simulated via the derivative of the particles’ current position, determining the gradient, much like in *gradient descent* via Taylor expansions in neural networks. The gradient steers the particle always towards the typical set. The particles’ momentum is additionally determined by its *potential energy* (the ”height” associated to its position in parameter space). When the particle starts ”sliding” towards a regions with high probability mass, its movement is similar than that of a marble rolling down a bowl. As in real physical systems, the potential energy gets converted to *kinetic energy* until it surpasses the lowest point, ”climbing up” the opposite curvature of the bowl. Now, kinetic energy gets re-converted into potential energy, as the particle climbs. A set of differential equations governs the ”sliding” behaviour, which is the result of the interplay of ”gravity”, ”potential energy” and ”kinetic energy” (Monnahan et al., 2017). This way, the particle (the markov chain), will always ”hover around” the typical set.

In HMC, the particles’ initial momentum is randomly determined. Then, its path is simulated and the end position on the parameter space is set as the new sampling point. For simple parameter spaces, the particles path is easily determined by the mentioned differential equations, but for high-dimensional spaces, the path has to be approximated by an numerical method, called *leapfrog-integrator*. This method itself has parameters for path approximation: the *step-size*, with similar practical repercussions as with the standard deviation in the Metropolis’ Gaussian proposal distribution. Should the step size be set too short, many ”leapfrog” integrations will be required until a transition occurs. Should it be too high, the particles’ path will overshoot its optimal position and it will have to turn back. In addition, inadequate step-sizes might lead the particles momentum to be set to infinity, which means that the path-approximation has failed. This error is called *divergent transition*. HMC, while improving sampling efficiency, does not eliminate the need for hand-tuning parameters. Small step sizes yield more accurate path approximations and thus, less overshooting and less proposal rejections, but also increase computation time (Monnahan et al., 2017). Because of this, optimizing step-size is crucial (Betancourt and Girolami, 2015). The No-U-Turn-Sampler (NUTS) eliminates the need for setting a step-size by piece-wise simulating the particles’ path and adjusting that parameter on-the-go.

4 Implementation of probabilistic models

The implementation and computation of probabilistic models is aided by probabilistic programming frameworks, such as *Stan* (Carpenter et al., 2017). In Stan, models that have a probabilistic definition can be expressed via probabilistic objects and be operated upon. Once the model is defined, the user specifies how many samples the HMC or NUTS sampler should run and which portion should be used as ”warm-up” or ”burn-in”. Warm-up samples are not being considered after all samples have been made, but help to get the sampler to a good starting point. This is specially important for the NUTS sampler, as a good step size will most likely only be calibrated after some 500 iterations. This might vary depending on the data set and complexity/parametrization of the model. The probabilistic model from Equation 2, formulated in the Stan language, will look like follows:

```
data {
  // This model only allows 1 measurement time
  int<lower=0> TI;
  // Mean RR over all measurement times
  int<lower=0> RR;
}

parameters {
    real d;
    real<lower=0> a;
    real<lower=0> sigma;
}

transformed parameters {
    real decay;
    decay = pow(TI, -d) - a // Our decay model
}

model {
    d ~ gamma(0.1, 0.01);
    a ~ normal(0.2, 0.1);
    sigma ~ gamma(0.1, 0.0)

    RR ~ normal(decay, sigma) // Likelihood calc.
}
```

For more information and details regarding model implementation, the over 500 page long Stan manual, containing statistical model examples and a more technical description of algorithms, methods and Stan modeling language can be found here: https://github.com/stan-dev/stan/. Stan automatically offers several sampling diagnostics to help the user asses if the posterior exploration has been successful. *Potential Scale Reduction* not only tests if different chains have arrived at the same mean, but also splits each markov chain samples in half and tests if each half has the same mean as the other. The sample distribution from any chains should be similar no matter if taken from the initial sampling or at the end (Carpenter et al., 2017), (Gelman et al., 2013). Potential scale reduction is denoted as $\hat{R}$ and it is recommended that it should never exceed the value of 1.1, anything else point to unbalanced, non-stationary sampling distributions and that the parameter space has likely not been correctly explored. The *effective sample size* N_{eff}, is a indicator of how auto-correlated the samples are. The more correlated samples in a markov chain are, the more are required to reach the same level of information diversity (Monte Carlo error) for an estimate. The way to interpret N_{eff} is in relationship to the total number of samples taken. If N_{eff} is only a small portion, then the samples suffer from strong auto-correlation. *Traceplots* depict time-series of the sampling iterations. While it is widespread practice to asses convergence with traceplots, Gelman et al. (2013) warns against this practice, as they do not scale well with many parameters. Instead, traceplots should only be considered after taking N_{eff} and $\hat{R}$ into account. *Divergent transitions* have been already discussed. They represent major failures in over-estimating step-size, which leads to biased parameter estimation as not the entire posterior is sampled from. While NUTS automatically estimates the best number of steps for approximating path length per iteration, on default, the maximum steps is set to 10 each iteration. This parameter is called *Maximum tree depth*. Should the sampler message that the maximum tree depth is often exceeded, only sampling efficiency is affected, samples will not be more biased because of this.

References

Betancourt, M. (2016). Bayesian modeling techniques in stan. Tokyo Stan.

Betancourt, M. (2017). A conceptual introduction to hamiltonian monte carlo. *arXiv preprint arXiv:1701.02434*.

Betancourt, M. and Girolami, M. (2015). Hamiltonian monte carlo for hierarchical models. *Current trends in Bayesian methodology with applications*, 79:30.

Carpenter, B., Gelman, A., Hoffman, M. D., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M., Guo, J., Li, P., and Riddell, A. (2017). Stan: A probabilistic programming language. *Journal of statistical software*, 76(1).

Dietz, L. (2010). Directed factor graph notation for generative models. *Max Planck Institute for Informatics, Tech. Rep.*

Gabry, J., Simpson, D., Vehtari, A., Betancourt, M., and Gelman, A. (2019). Visualization in bayesian workflow. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 182(2):389–402.

Gelman, A. and Hill, J. (2006). *Data analysis using regression and multilevel/hierarchical models*. Cambridge university press.

Gelman, A., Simpson, D., and Betancourt, M. (2017). The prior can often only be understood in the context of the likelihood. *Entropy*, 19(10):555.

Gelman, A., Stern, H. S., Carlin, J. B., Dunson, D. B., Vehtari, A., and Rubin, D. B. (2013). *Bayesian data analysis*. Chapman and Hall/CRC.

Griffiths, T., Kemp, C., and Tenenbaum, J. (2008). Bayesian model of cognition. In Sun, R., editor, *Cambridge Handbook of Computational Cognitive Modeling*, chapter 1, pages 1–47. Cambridge University Press.

Katahira, K. (2016). How hierarchical models improve point estimates of model parameters at the individual level. *Journal of Mathematical Psychology*, 73:37–58.

Kruschke, J. (2014). *Doing Bayesian data analysis: A tutorial with R, JAGS, and Stan*. Academic Press.

Lee, M. D. (2011). How cognitive modeling can benefit from hierarchical bayesian models. *Journal of Mathematical Psychology*, 55(1):1–7.

Love, B. C., Ramscar, M., Griffiths, T. L., and Jones, M. (2015). Generative and discriminative models in cognitive science. In *CogSci*.

Melnik, R. (2015). *Mathematical and computational modeling: with applications in natural and social sciences, engineering, and the arts*. John Wiley & Sons.

Monnahan, C. C., Thorson, J. T., and Branch, T. A. (2017). Faster estimation of bayesian models in ecology using hamiltonian monte carlo. *Methods in Ecology and Evolution*, 8(3):339–348.

Ng, A. Y. and Jordan, M. I. (2002). On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Advances in neural information processing systems*, pages 841–848.

Pearl, J. (2009). *Causality: models, reasoning, and inference*. Cambridge University Press, 2nd edition.

Shiffrin, R. M., Lee, M. D., Kim, W., and Wagenmakers, E.-J. (2008). A survey of model evaluation approaches with a tutorial on hierarchical bayesian methods. *Cognitive Science*, 32(8):1248–1284.