Università degli Studi di Padova

Department of Mathematics 'Tullio Levi-Civita'

Master of Science in Data Science

Variational Inference: Recent Advances and Applications to Dynamic Ranking on Graphs

Supervisor: Marco Formentin

Department of Mathematics

Co-Supervisor: Caterina De Bacco

Max Planck Institute for Intelligent Systems

Student: Nicoló Ruggeri

N. 1178646

Academic Year 2018/2019

To my girlfriend Alessandra,
for giving me more than
I could ever ask for.

Contents

1. Introduction

When fitting statistical models to data, often the main goal is to estimate a set of parameters that allows a description of the reality as faithful as possible. Thanks to the increasing computational power of modern computers, as well as availability of large amounts of data, increasingly complex statistical and machine learning algorithms are trying to extract as much information as possible for myriads of applications.
Probabilistic models are part of these. Generally speaking, probabilistic models are abstractions of reality that try to find patterns in the data at hand. When building these models, one often relies on incorporating some structure that is believed to be rich and descriptive of some underlying phenomena. Usually this structure depends on some *parameters* or on some *latent variables*.
Broadly speaking, a classic distinction for modelling uncertainty is given by the choice between frequentist and bayesian approaches. In the latter case, the inference procedure boils down to transforming some prior beliefs, that the modeller has on reality, into posterior ones, interpolating between prior knowledge and observed data. Believes are in this case encoded by probability distributions over the world, both observable and not. In other words:

"Bayesian inference is the process of fitting a probability model to a set of data and summarizing the result by a probability distribution on the parameters of the model and on unobserved quantities such as predictions for new observations" [1].

However, posterior inference is not always easy, and in many, not necessarily complex scenarios, retrieving an analytical solution is impossible. For this reason, the classical solution is to resort to *approximate inference* techniques, where we abandon the search for an exact distribution and turn instead to finding some "good" approximations. In this framework many algorithms have been created, many of which stemming from classical statistical physics solutions to similar problems. Nowadays, this is a very active branch of research, due to the importance and

increasing applicability of bayesian models/probabilistic machine learning.
Among the available approximate inference techniques, a lot of interest has been given to *variational inference* [2, 3], due to its relative light computational weight and to some recent advances that make the approach quite flexible to address a wide range of applications.
For this thesis we set two goals. Firstly, contextualize variational inference in a broader inference perspective. Then, dive into the most recent advances in the field, thus giving a presentation as complete as possible of the research questions still open. Secondly, we present an original approach to dynamic ranking on graphs, which uses some recent techniques of variational inference to improve the state of the art in this particular setting.

Structure

The thesis will be divided as follows. Section 2 gives a brief introduction about inference, starting from the differences between frequentist and bayesian inference. We then proceed by presenting some of the classic approximate inference techniques. Notice that this short section does not aim at providing an exhaustive presentation about these themes, each of which would require an entire book on its own. Rather, we aim at giving a broader context to have a better idea of why variational inference is useful, and to compare with different more classical solutions.
In section 3 we present variational inference (VI), starting from the basics and then moving to the most recent advances. To provide a complete digression, different techniques for richer and more effective VI are presented, along with some relevant applications.
Finally, in section 4 we present VariationalDynamicRanking (VDR), a novel approach to dynamic ranking on graphs based on bayesian modelling techniques and variational inference approximations.

2. Inference, a general framework

We begin our introduction with a classic example, comparing the linear regression model in a frequentist and bayesian setting. This is needed to set the first notations, as well as presenting some cases where explicit analytical calculations are possible, before proceeding with more complex cases.

2.1 Frequentist and Bayesian inference

The difference between the frequentist and bayesian approach boils down to the very definition of probability. This discussion lead to profound differences in treatment of data, and many times to ideal divergences between statisticians [4, 5]. For the sake of our discussion, the main difference between frequentist and bayesian statistic is relative to modelling. The frequentist approach assumes that there are some "true" parameters underlying the phenomena under study, and tries to discover them using inference. Under this view, uncertainty is due to the data. The bayesian approach, in turn, is to think of the observed phenomena as due to some underlying distribution of parameters, and tries to model and extrapolate the distribution of this parameters using the data at disposal. Under this view then, uncertainty comes from the parameters themselves, while data are given [1].

Before diving into some practical examples, it is useful to set some notations and definitions. We will use them extensively throughout the following chapters.

Definition 2.1. Given a set of **observations** $(x_i, y_i)_{i=1,\dots,m}$, we define as **covariates** $x_i \in \mathbb{R}^n$ and as **dependent** (or **response**) **variables** the $y_i \in \mathbb{R}$. Aggregating the data we obtain an **observation matrix** $X \in \mathbb{R}^{m,n}$, containing as rows the covariates x_i^T, and **outcome vector** $y = (y_1, \dots, y_m)^T \in \mathbb{R}^m$.

Definition 2.2. Given a set of data $(x_i, y_i)_{i=1,\dots,m}$ and a vector of parameters θ, we define the **likelihood** of the data *given* the parameters as the probability

$p(y|X;\theta)$[1].

2.1.1 Example: linear regression

Frequentist linear regression

The classic frequentist setting for linear regression is the following:

$$y = X\beta + \epsilon \tag{2.1}$$

$$\epsilon \sim \mathcal{N}(0, \sigma^2 \mathbb{1}) \tag{2.2}$$

Usually, one wants to determine the best regression parameters β given the data. This implies specifying what we mean by "best" coefficients. One standard solution to this problem is the maximum likelihood approach [6]. This means finding the coefficients β given by

$$\underset{\beta}{\operatorname{argmax}}\, p(y|X;\beta)\,,$$

i.e. the coefficient that maximize the likelihood of the data.
Under the assumption of gaussian distributed error (2.2), the maximization of the likehood is achieved in closed form:

$$\hat{\beta} = (XX^T)^{-1}X^T y \tag{2.3}$$

For explicit calculations, see [7].

Bayesian Linear Regression

The formulation of the classic bayesian linear regression is similar to the frequentist one, but with a different formulation of uncertainty. We mainly base our presentation on [8], but several other textbooks can be found in the literature.
As we described above, in a bayesian setting one tries to capture uncertainty in the model itself, rather than in the data observed. The technical way to represent this is to think of the parameters of a model as random variables themselves, and assign a prior distribution to them as well. In the following we introduce some common building blocks needed to better characterize this framework.

[1]Notice that here we are not interested in modeling the probability of the covariates X explicitly, but rather the probability of the outcome y given some covariates. For this reason we condition on X rather than considering $p(y, X|\beta)$. Also, covariates may not always be available in settings different from regression; in those cases we can drop altogether the conditioning on X.

Definition 2.3. Consider some model parameters θ and assume that they follow a certain probability distribution $p(\theta)$. Given these model parameters, we assume that the observations follow a certain distribution $p(y|x,\theta)$. We now give a name to the following distributions:

$p(\theta)$	**prior** distribution
$p(y\|X,\theta)$	**likelihood**
$p(\theta\|X,y)$	**posterior** distribution
$p(y\|X)$	**evidence** or **marginal likelihood**

Performing inference in this context means to *find the posterior distribution*, i.e. characterizing the source of uncertainty in a quantitative way, given the observed data.

Given these definitions, we can now describe the standard bayesian linear regression model. From now on, we assume that the noise variance σ^2 is known.

- Likelihood: similarly to the frequentist formulation, $p(y|X,\beta) = \mathcal{N}(X\beta, \sigma^2\mathbb{1})$, where $\mathcal{N}(\mu,\Sigma)$ is the gaussian density with mean μ and covariance Σ. Notice however that, differently from the frequentist case, here we *condition* on the parameters β, as they are considered random variables in this framework, rather than fixed parameters. We highlight this distinction using two different notations: $p(y|X,\beta)$ if we *condition* on β and $p(y|X;\beta)$ when β is a deterministic parameter.

- Prior: we assign a gaussian prior $p(\beta) = \mathcal{N}(\beta_0, \Sigma_0)$.

Our goal now is to find the posterior distribution. We can do so by using Bayes theorem [1] and noticing that $p(\beta|X) = p(\beta)$:

$$\begin{aligned} p(\beta|X,y) &= \frac{p(y|X,\beta)p(\beta)}{p(y|X)} \\ &= \frac{\mathcal{N}(X\beta,\sigma^2\mathbb{1})\mathcal{N}(\beta_0,\Sigma_0)}{p(y|X)} \quad . \end{aligned}$$

Typically, one does not need to compute the normalizing factor $p(y|X)$: if we are able to find a known distribution in the nominator $p(y|X,\beta)p(\beta)$, then we can automatically find it as $p(y|X) = \int p(y|X,\beta)p(\beta)d\beta$. For this reason we usually

write the calculations above as:

$$\begin{aligned} p(\beta|X,y) &\propto p(y|X,\beta)\, p(\beta) \\ &= \mathcal{N}(X\beta, \sigma^2 \mathbb{1})\, \mathcal{N}(\beta_0, \Sigma_0) \quad . \end{aligned}$$

Notice that the product of two gaussian densities is also a (unnormalized) gaussian density, so that

$$p(\beta|X,y) = \mathcal{N}(\beta_p, \Sigma_p)\,,$$

with

$$\begin{aligned} \beta_p &= \Sigma_p^{-1}\Sigma_0\beta_0 + \frac{1}{\sigma^2}\Sigma_p X^T y \\ \Sigma_p &= \sigma^2(\sigma^2\Sigma_0^{-1} + X^T X)^{-1}\,, \end{aligned}$$

where the inverse Σ_p^{-1} can be explicitly found via matrix inversion lemma [9]. Notice that this computation involving gaussians were easy due to the nice properties of these distributions, but this is often not the case.
For example, assuming that the noise variance σ^2 is not known things become more complicated. In this case, assuming also a normal prior on σ^2 would not lead to a tractable posterior form; in general, one would need to consider less common distributions. However, a tractable posterior is obtained by using a prior following a *normal-inverse-gaussian* distribution. In fact, considering again a gaussian likelihood, it can be proven [8] that the corresponding posterior is a normal-inverse-gaussian.

2.1.2 Bayesian inference: conjugate priors and exact inference

In the second example of section 2.1 we saw a situation where, given a certain combination of prior and likelihood, the posterior could be determined in closed form. In general, we can use Bayes rule to express the posterior as:

$$\text{posterior} = \frac{\text{(likelihood) x (prior)}}{\text{evidence}} \propto \text{(likelihood) x (prior)} \tag{2.4}$$

In general, starting from an arbitrary combination of likelihood and prior would not lead to a tractable expression for the posterior. For a concrete example where this actually happens see section 2.2.1. However, in the previous example we saw that, with the particular choice of gaussian likelihood and prior, we were able to obtain

a gaussian posterior distribution. This is an example of a tractable posterior. To characterize these types of scenarios we introduce the following definition:

Definition 2.4. Given a family of likelihood distributions $p(y|\theta) \in \mathcal{H}$, we say that the family of priors $p(\theta) \in \mathcal{P}$ is **conjugate** with respect to the family $\mathcal{H}$ if the resulting posterior still belongs to $\mathcal{P}$.

Given this definition, we saw that the gaussian family is conjugate with itself, but more complex conjugacy relationships exist. For example bernoulli likelihoods have a conjugate beta prior and uniform distributions have conjugate pareto priors. A complete table of conjugate distributions is available at [10].
While performing exact inference using conjugate priors requires only the computation of the posterior parameters from the models' ones, in many relevant applications conjugacy does not hold. In this cases, **approximate inference methods** need to be used. In the following chapters we focus on presenting some of these.

2.2 Approximate inference

To motivate our interest towards approximate inference, and to show that conjugacy could not hold in fairly basic models, we start with a simple example. The following presentation is based mainly on [8, 11].

2.2.1 Example: bayesian logistic regression

We introduce here a standard logistic regression model in a bayesian setting. As for the frequentist case, we can think of logistic regression as a linear model with a sigmoid activation applied over the regressed values. Formally, using the notation from the previous sections, we consider binary observations $y \in \{0, 1\}$. We model this as:

$$\begin{aligned} y &\sim Be\left(\sigma(X\beta)\right) \\ \beta &\sim \mathcal{N}(\beta_0, \Sigma_0)\,, \end{aligned}$$

where σ is the sigmoid function and $Be(\cdot)$ denotes the Bernoulli distribution.

By proceeding using Bayes rule we notice that equation 2.4 is analytically intractable, due to the integral given by

$$\text{evidence} = p(y) = \int p(y|\theta)p(\theta)d\theta \quad .$$

We are therefore forced to find alternative ways to restore the posterior distribution of the model's parameters.

2.2.2 Laplace approximation

The first method that we present is the Laplace approximation method. This technique tries to find a gaussian approximation of the posterior in the following way. Consider the posterior distribution of the parameters $p(\theta|y)$. In analogy with statistical physics methods and the Boltzmann distribution with a given energy function [12], we define the **energy function**

$$E(\theta) := -\log p(\theta|y) + const\,,$$

and we can rewrite the posterior as

$$p(\theta|y) = \frac{1}{Z} e^{-E(\theta)} \tag{2.5}$$

for some normalization factor Z. In general, deriving a tractable expression for Z is a highly non trivial problem, as it depends on the integral of the energy, which in turn depends on the unknown posterior.

In developing our approximation, we center it on the maximum a priori estimator θ^*, i.e. the minimum of $E(\theta)$. This choice is due to two considerations. First, we expect that most of the mass of $p(\theta|y)$ in 2.5 will be centered around θ^*, as it has the lowest energy. Second, the analytic property that $\nabla E_{|_{\theta^*}} = 0$, being the minimum a stationary point. We therefore perform a Taylor expansion of E around θ^* to find:

$$E(\theta) \approx E(\theta^*) + \frac{1}{2}(\theta - \theta^*)^2\, \nabla^2 E_{|_{\theta^*}} (\theta - \theta^*)$$

and we retrieve the *approximate* posterior [8] as:

$$p(\theta|y) \propto e^{-\left[E(\theta^*) + \frac{1}{2}(\theta-\theta^*)^2 \nabla^2 E_{|_{\theta^*}} (\theta-\theta^*)\right]} = \mathcal{N}(\theta; \theta^*, (\nabla^2 E_{|_{\theta^*}})^{-1})\,.$$

Due to the central limit theorem, asymptotic distributions tend to be approximately gaussian, making this method more reliable.

2.2.3 Markov chain Monte Carlo approximation

The classical way of approximating and sampling from complex distributions is given by Markov chain Monte Carlo (MCMC) methods. At their core these methods rely on trying to find a Markov chain whose invariant distribution is the one that we want to sample from. While the two reference algorithms are the Metropolis-Hastings and Gibbs sampling, the world of MCMC is broad and covering it is out of the scope of this thesis. For an introduction we refer to [11] and for a thorough presentation, with applications to bayesian inference, to [13].
For our scopes two features of MCMC methods are especially relevant. Importantly, they have asymptotic convergence guarantees, making them the gold standard for sampling from complex distributions. However, the major drawback is given by their very slow convergence. In typically high-dimensional applications, such as the ones from modern machine learning, these methods become unfeasible also with great computational power. This is one of the main reasons why the methods that we are going to present are gaining increasingly more popularity in the Machine Learning community.

3. Variational Inference

In the previous chapter we introduced some of the classical methods for approximate inference in cases where the exact posterior is not available. Here instead we focus on a new methodology, Variational Inference [2], that has received a wide recognition in the statistics and machine learning communities.
In the following sections, we will present the theory underpinning the algorithmic developments. We then present some recent results in this actively investigated field. Finally, we will show some applications and open lines of research.
We start by presenting what we call the **fundamental theorem of variational inference**, following the proof from [14].

Theorem 3.1. *Consider a model with observed variables $x \in \mathcal{X}$ and latent variables $z \in \mathcal{Z}$. Then for any probability distribution q the following formula holds:*

$$\log p(x) = KL(q(z) \,||\, p(z|x)) + \mathcal{L}(x, p, q) \tag{3.1}$$

where $KL(q(z|x)||p(z|x))$ is the Kullback-Leibler divergence between the two distributions and

$$\mathcal{L}(x, p, q) := \mathop{\mathbb{E}}_{z\sim q(z)}\left[\log \frac{p(x,z)}{q(z)}\right] \tag{3.2}$$

Proof.

$$\begin{aligned}
\log p(x) &= \mathop{\mathbb{E}}_{z\sim q(z)} \log p(x) \\
&= \mathop{\mathbb{E}}_{z\sim q(z)} \log \left(\frac{p(x,z)}{p(z|x)}\right) \\
&= \mathop{\mathbb{E}}_{z\sim q(z)} \log \left(\frac{p(x,z)}{q(z)}\frac{q(z)}{p(z|x)}\right) \\
&= \mathop{\mathbb{E}}_{z\sim q(z)} \log \frac{p(x,z)}{q(z)} + \mathop{\mathbb{E}}_{z\sim q(z)} \log \frac{q(z)}{p(z|x)} \\
&= \mathcal{L}(x, p, q) + \mathrm{KL}(q(z) \,||\, p(z|x))
\end{aligned}$$

□

Definition 3.2. The term $\mathcal{L}(x, p, q)$ in (3.1) is called **Evidence LOwer Bound**, in short **ELBO**. The name comes from the fact that, since the KL-divergence is always non-negative, $\mathcal{L}(x, p, q)$ is a lower bound of the log-evidence $\log p(x)$.

This theorem, and the consequent definition of ELBO, are very important due to the following observation:

$$\operatorname*{argmin}_q \mathrm{KL}(q(z) \,||\, p(z|x)) = \operatorname*{argmax}_q \mathcal{L}(x, p, q)\,, \tag{3.3}$$

as the term $\log p(x)$ does not depend on q.
This means that choosing the distribution q within a given family such that it minimizes the KL distance with the real posterior, is equivalent to minimizing the ELBO. This is useful because, while we do not know the real posterior $p(z|x)$, and therefore the value of $\mathrm{KL}(q(z) \,||\, p(z|x))$, we are nonetheless, in principle, able to compute the ELBO, as this only depends on the joint distribution $p(z, x)$ and q, which we know analytically.
In the Variational Inference (VI) terminology, the distribution q is called **variational approximation**, and is usually parametrized by some parameters $\theta \in \mathbb{R}^d$. Therefore "variational inference turns the inference problem into an optimization problem" [3]

$$\operatorname*{argmax}_\theta \mathcal{L}(x, p, q_\theta) \quad . \tag{3.4}$$

After the explicit computation of the ELBO (3.2), which entails the computation of expected values with respect to the variational distribution, one just needs to run any arbitrary optimization method, typically (conjugate) gradient descent over the parameters θ governing q. Notice, however, that the explicit computation of the ELBO is not always straightforward; alternative methods have been developed when this is not analytically available, see section 3.2.

Example 3.3. A nice example where the ELBO is analytically available is given by the logistic regression presented in section 2.1.2. Complete and generalized computations can be found in [15].

3.1 Mean field variational inference

One of the main issues faced when using variational methods, is the choice of the variational approximation q. Ideally, the variational approximation family should

belong to a family $\mathcal{F}$ rich enough that the KL divergence

$$\mathrm{KL}(\mathcal{F} \,||\, p(z|x)) = \min_{q \in \mathcal{F}} \mathrm{KL}(q(z) \,||\, p(z|x))$$

is low enough, while still allowing tractable computations of the ELBO.
One traditional choice is the so called **mean field approximation**. This means choosing a factorized variational approximation

$$q_\theta(z) = \prod_i q_{\theta_i}(z_i)\,,$$

where every q_{θ_i} belongs to a distributions family of choice. While at first sight this may look like an oversimplification, it works in practice as it is the predominant choice in many applications, see section 3.4.3. Further ideas for structured mean field and more involved distributions are still under investigation. We will cover some of the recent advances in section 3.3.
Notice that using a factorized distribution means neglecting, at least in first approximation, all the possible correlations in the posterior distribution. While this is similar to other machine learning scenarios, for example naïve bayes classifiers [6], we can see that, in mean field variational inference, correlations still play a role in determining the optimal θ^*. This is because, even if in the variational approximation variables are decorrelated, the KL is minimized to approximate the posterior, where correlations can in principle be present.

3.1.1 Mean field, exponential families and coordinate ascent

Some optimization procedures for mean field involving the so called *conditionally conjugate exponential families* have been developed in the past years. A great review on this topic is represented by [3]. These procedures use tools from calculus of variations and is at the origin of the variational inference's name. However, these procedures, based on closed from updates for coordinate ascent, are restricted to a small class of families, leaving optimization in the more general case an open issue. We present the most modern and general algorithm in section 3.2. The interested reader can find in [3] a relevant example of a mixture of gaussians where the variational framework is implemented via the coordinate ascent "CAVI" algorithm.

3.1.2 Amortized inference

Amortized variational inference tries to reduce the number of parameters in the variational approximation by specifically introducing the observable data (i.e. the covariates) into the approximation itself. Formally, the form of the variational approximation is

$$q_{\theta(x)}(z)\,,$$

where now $\theta(x)$ is a function depending on some parameters to optimize. As we can see here the problem of optimizing the parameters is a regression one. The term *amortized* comes from the fact that, given z and x, we can model them as above instead of being forced to run an iterative algorithm to optimize a different set of parameters for $q_i(z_i|x_i)$. The idea of amortized inference is at the base of the connection between deep learning and variational inference, which we will briefly introduce in section 3.4.2.

3.2 Black Box Variational Inference

The mean field approximation, in general, may not be enough to derive analytically all the terms involving expectations inside the expression for the ELBO (3.2). In these cases, one possibility is to optimize using (stochastic) gradient descent methods [16, 17, 18]. However, it is often prohibitive to compute terms like

$$\nabla_\theta \mathbb{E}_q[\mathcal{L}(x,p,q)]\,, \tag{3.5}$$

as the parameters θ are not only inside the argument of the expectation, but also in the distribution with respect to which the expectation is taken, since $q = q_\theta$. Therefore

$$\nabla_\theta \mathbb{E}_q[\mathcal{L}(x,p,q)] \neq \mathbb{E}_q[\nabla_\theta \mathcal{L}(x,p,q)]\,.$$

This is unfortunate as the right-hand-side of this expression is usually easier to exploit algorithmically. One option would be to find a stochastic approximation of the gradient and then calculate its expectation, for instance, by sampling from q.

One easy way to overcome this problem is the so called "reparametrization trick" [19], which is at the base of deep learning applications of variational inference 3.4.2. In short, this trick allows to backpropagate gradients also on the expectation $\mathbb{E}_q$. We show how this works for obtaining a stochastic approximation $\tilde{\nabla}_\theta \mathcal{L}$ of the gradient of the ELBO with an example. Consider $q = \mathcal{N}(\mu, \sigma)$, and thus $\theta = (\mu, \sigma)$.

We then sample $s_i \sim \mathcal{N}(0,1)\,, i = 1, \ldots, n$ and use the expression $z_i = \sigma s_i + \mu$ to compute

$$\begin{aligned}\tilde{\nabla}_\theta \mathcal{L} &= \frac{1}{n}\sum_{i=1}^{n} \nabla_{(\mu,\sigma)}(\log p(x, z_i) - \log q_{(\mu,\sigma)}(z_i)) \\ &= \frac{1}{n}\sum_{i=1}^{n} \nabla_{(\mu,\sigma)}(\log p(x, \sigma s_i + \mu) - \log q_{(\mu,\sigma)}(\sigma s_i + \mu))\,.\end{aligned}$$

As we can see, here we approximate the real, in general non analytic gradient from 3.5, with a monte carlo estimation given by an average of n explicitly computable gradients. Automatic differentiation softwares, e.g. *Pytorch* [20], provide ready-to-use implementations of this trick. Even though there have been recent advances to generalize this approach [21], it is still not applicable in general. In particular, this does not work for discrete distributions.

A more general solution has been proposed in [22]. They develop a "black-box" methodology that, for any q and p, only requires estimating $\nabla_\theta \log q_\theta$. With these ingredients, one can obtain a stochastic gradient approximation that can be then passed as input into any gradient descent algorithm. We report here only the main result, and refer the reader to the original paper for the full proof.

Theorem 3.4. *Assuming that $q_\theta(z)$ is differenciable in θ, the following relationship holds:*

$$\begin{aligned}\nabla_\theta \mathcal{L}(x, p, q) &= \nabla_\theta \mathbb{E}_q[\log p(x, z) - \log q(z)] \\ &= \mathbb{E}_q[\nabla_\theta \log q_\theta(z)(\log p(x, z) - \log q_\theta(z))]\,. \qquad (3.6)\end{aligned}$$

Using this result allows for simple optimization, presented in algorithm 1

Algorithm 1 Black-Box Variational Inference

1: **Choose** model p, variational approximation q, n batch size, ρ_t (dynamic)step size for optimization
2: **while** ELBO has not converged **do**
3: sample n latent variables from q: $\{z_i\}_{i=1,\ldots,n} \sim q_\theta(z)$
4: compute stochastic approximation
$$\tilde{\nabla}_\theta \mathcal{L} = \tfrac{1}{n}\textstyle\sum_{i=1}^{n} \nabla_\theta \log q_\theta(z_i)(\log p(x, z_i) - \log q_\theta(z_i))$$
5: update variational parameters $\theta = \theta + \rho_t \tilde{\nabla}_\theta \mathcal{L}$

This approach is not immune to problems though. In fact if one would be able to compute the ELBO in closed form, the derivatives would be free from any type of noise and optimization could proceed smoothly. In black-box variational

inference instead, we are just able to recover a stochastic, unbiased approximation of the gradient. This is typically affected by a very large variance, making the search for good local optima very hard in many applications. The problem is so severe that in the original paper itself two methods for variance reduction are proposed: Rao-Blackwellization of the gradient and the use of control variates. We refer to [22] for further details.

3.3 Beyond mean field

While the use of mean-field VI is predominant due to its computational convenience, as well as ease of implementation, many efforts have recently been made towards using techniques allowing for more refined variational approximations. In a series of recent papers several results proved the benefits of better approximating the exact posterior. Here we briefly present some of these advances.

3.3.1 Structured mean field

The first idea towards moving away from vanilla mean-filed approximations is to use variational distributions that capture some structure, while still preserving a factorized form. One can then view the parameters θ_i and variables z_i as divided in blocks, where the structure allows to still perform inference [8]. This framework is quite general but still imposes strong assumptions, limiting the approximation abilities of the distribution q. We present an example in our work from section 4.2.3.

3.3.2 Normalizing flows

The idea of using *normalizing flows* has been introduced in [23, 24, 25]. When dealing with variational approximation one always has to pay some attention towards building scalable methods. This is in fact the main strength of variational inference, that bear no convergence guarantee, with respect to the asymptotically exact MCMC.
The idea of normalizing flows is to then try and start from a simple and cheap mean field family, making it more complex and expressive using some specific invertible transformations. Following the notation from [23], the technical framework is given by considering a sequence of invertible and differenciable transformations $f_1, \dots, f_K$. Starting from an initial mean field approximation $z_0 \sim q_0 = \prod_i q_{0,i}(z_i|\theta_i)$ one considers the "flow" of transformations $z_i = f_i(z_{i-1})$ that grow

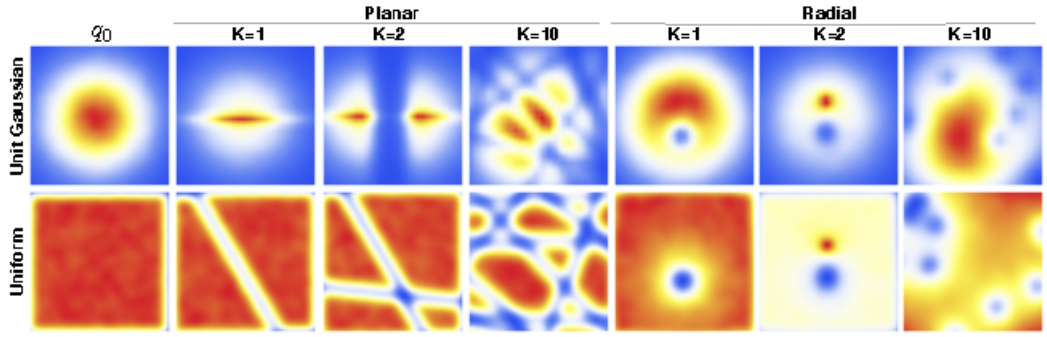

Figure 3.1: starting from simple distributions (unit gaussian and uniform) one can see that applying an increasing number K of normalizing flows leads to very rich and multimodal distribution. Figure taken from [23].

more and more complicated till the final approximation $z_K \sim q_K$.

While it is clear that considering complicated and non uncorrelated distributions q_K should ideally lead to better approximation, see figure 3.1, one also has to prove that inference stays cheap, ideally linear in the number of parameters. This is achieved by considering the pdf of the transformed variables, which depends on the original via the Jacobian $\frac{\partial f_k}{\partial z_{k-1}}$

$$\log q_K(z_K) = \log q_0(z_0) - \sum_{k=1}^{K} \log \left| \det \frac{\partial f_k}{\partial z_{k-1}} \right| .$$

Given this formula we can sample and optimize without even knowing the final distribution q_K explicitly. One just needs to sample from the (cheap and factorized) initial distribution q_0 and then follow the flow $f_K \circ \ldots \circ f_1$ to find a realization of z_K. For optimization we can use simple black box methods, as from section 3.2. To find the ELBO we use the law of the unconscious statistician

$$\mathbb{E}q_K[h(z_K)] = \mathbb{E}_{q_0}[h(f_K \circ \ldots \circ f_1(z_0))]$$

and find

$$\begin{aligned} \mathcal{L} &= \mathbb{E}_{z_0 \sim q_0}[\log q_K(z_K) - \log p(x, z_k)] \\ &= \mathbb{E}_{z_0 \sim q_0} \left[\log q_0(z_0) - \sum_{k=1}^{K} \log \left| \det \frac{\partial f_k}{\partial z_{k-1}} \right| - \log p(x, z_k) \right] . \end{aligned}$$

As we can see this procedure keeps the efficiency of the usual black box methods,

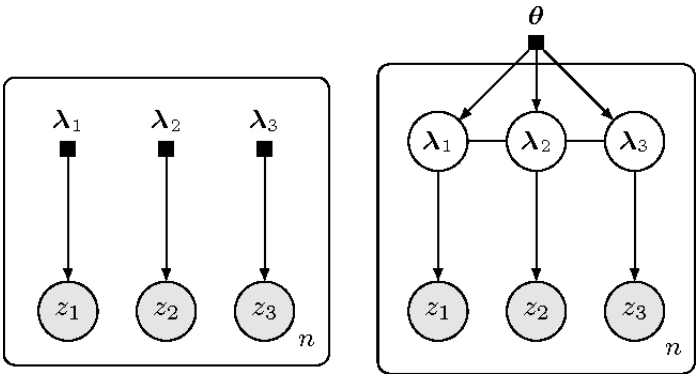

Figure 3.2: **left** the vanilla mean field approximation q. There is no correlation between the latent variables z. **right**: the new hierarchical model, where correlations are introduced by means of the new variational parameters θ. Picture taken from [28]

.

and requires linear time sampling and optimization in the length of the flow K. Some care has to be given to the form of the transformations f_i, as computing the Jacobian must be very cheap or ideally not needed. For example considering transformations with triangular Jacobians allows to compute the determinants by just multiplying the diagonal terms. A literature about proper and rich choices of these transformations has developed lately in response of the utility of this method. Examples are given by [26, 27].

3.3.3 Hierarchical models

Another scalable solution towards inserting correlation in variational methods is to insert hierarchical structures in the bayesian modelling itself [28]. While we usually model the posterior using a $q(z;\lambda)$ depending on some variational parameters λ, now we introduce an extra layer, assigning a prior to the λ, obtaining

$$q(z,\lambda;\theta) = q(z|\lambda)\, q(\lambda;\theta)$$

depending on the new variational parameters θ. While to preserve scalability we need to keep on using a factorized distribution on the z, given by

$$q(z,\lambda;\theta) = q(\lambda;\theta) \prod_i q(z_i|\lambda_i)$$

we can still introduce a level of correlation given by the new layer of the variational approximation (figure 3.2).

The new ELBO has the following form

$$\mathcal{L}(p, q, \theta) = \mathbb{E}_{(z,\lambda)\sim q_\theta}[\log(p(x, z)) + \log r(\lambda|z; \theta) - \sum_{i=1}^{d} \log q(z_i|\lambda_i) - \log q(\lambda; \theta)]$$

where $r(\lambda|z; \theta)$ is a recursive approximation term for the unknown $q(\lambda|z; \theta)$. Stochastic estimators of the gradient can be found via reparametrization trick (see section 3.2). Many families are proposed in the original paper as candidate variational priors $q(\lambda; \theta)$, such as mixtures of gaussians or normalizing flows (see section 3.3.2).

3.4 Variational Inference: applications

In this section we will focus on presenting some recent applications that arose from or are related to variational inference. We will then present an original application to dynamic ranking on graphs and the relative inference solution in section 4.

3.4.1 Variational EM

Expectation Maximization (EM) is an algorithm for finding maximum likelihood (ML) or maximum a posteriori (MAP) solutions in models involving latent variables z. The review of the EM algorithm falls outside the scope of this thesis, we refer to [11, 8] for a great introduction.
The EM procedure is divided in two steps, called E-step (expectation step) and M-step (maximization step). During the former, one needs to find the posterior distribution of the latent variables $p(z|x)$ and during the M-step a maximization with respect to the hyperparameters governing p is performed. EM in based on theorem 3.1 and, in its exact original form, requires a closed form E-step. However, this is not possible in many scenarios, forcing the use of approximate versions of the E-step.
Variational EM falls within this approximate EM class in a bayesian setting, where we substitute the real posterior $p(z|x)$ with the variational approximation $q(z)$. The optimization loop is then iterated alternating a variational E-step, where we find an optimal q given the current hyperparameters, and an M-step, where q is kept fixed and one tries to optimize the ELBO as a function of the hyperparameters.
As a final remark about variational EM, or EM in a general bayesian context, notice that the hyperparameters defining the model can be divided in two categories: the parameters related to the prior and the ones determining the shape of

the likelihood. While it is in principle possible to perform the M-step optimizing both of them, as is done in some cases, a pure bayesian approach would just allow optimization over the likelihood parameters. This is due to the so called *likelihood principle*, that states that all the information about the data should be contained in the likelihood. Therefore the prior distribution should be left untouched, as it doesn't need to be related to the data [29]. We will present an example of this procedure during the optimization of our original model for dynamic ranking in section 4.2.3.

3.4.2 Variational Autoencoders

Variational autoencoders (VAEs) [19, 30, 31, 14] exploit the ideas from sections 3.1.2 and 3.2 to connect deep learning and variational inference. At their essence, VAEs try to simultaneously optimize the likelihood and the variational approximation at the same time, where they both assume an amortized form

$$p(x|z) = p_{\phi(z)}(x)$$
$$q(z|x) = q_{\theta(x)}(z)$$

where the functions $\phi(x), \theta(z)$ contains some parameters to optimize. In typical VAEs the prior is assumed to be a standard normal distribution, even if some generalizations have been recently introduced [21]. The way these functions are actually implemented is using neural networks: $q_{\theta(x)}(z)$ will play the role of an *encoder* that, when given some data in input, will try to model the posterior distribution of the latent variables. $p_{\phi(z)}(x)$ instead, will play the role of a decoder that, given some z extracted from the prior, can produce some new data in a generative fashion. Practically, the whole model is optimized in one run as a typical encoder-decoder deep learning architecture, with the addition of a random draw from the prior $N(0, 1)$ for every data point. The final result is in principle a model able to generate new unseen data exploiting the structure learnt from the decoder, by simply drawing from the prior whenever a new realization is needed. Moreover, given the simple structure of the prior, one is able to explore the latent space actively to qualitatively check how different features of the data are learnt and embedded into the space.

VAEs are one of the major variational inference applications and have become a very research intensive topic in recent years. We limit to highlight a very simple fact that is currently at the heart of many questions in VAEs research. In traditional variational inference, no mention is given to the parameters governing

the prior and likelihood of the model. This because they are kept fixed, and the approximation q is modified so as to be as close as possible, in KL divergence, to the posterior p. In VAEs however, also the parameters of the likelihood itself are modified for every pass of backpropagation through the neural network. This doesn't comply with the usual optimization procedure, as it produces an ever moving posterior p, which makes an unclear objective for the optimization of q. While this fact is often left unmentioned, it is very important as it totally changes the interpretation of the performed optimization. A very recent and interesting line of research is the one trying to merge the idea of normalizing flows, building towards richer approximations, see section 3.3.2, with VAEs. Some examples are given by [32, 33, 34].

3.4.3 Other applications

In principle the variational methods can be useful whenever the posterior distribution in a bayesian context is not available explicitly. For this reasons many recent works leverage this feature for performing inference in complex scenarios. Some examples relate to community detection in networks [35], static [36] and dynamic [37] topic models, document classification [38], recommender systems [39], reinforcement learning [40] and many others.

4. Variational Dynamic Ranking

4.1 Tools and notations

After having introduced the variational inference framework, we present here an application to an original field, namely, dynamic ranking on graphs. First, we formally define each of these three terms separately.

Definition 4.1. A **graph** G is a tuple $G = (V, E)$ composed of a *vertex* (or *node*) *set* $V = \{1, \dots, n\}$ and an *edge set* $E \subseteq V \,\mathrm{x}\, V$. A graph is said to be *undirected* if

$$(i, j) \in E \iff (j, i) \in E$$

i.e. if an edge $(i\, j)$ is present, also (j, i) is. A non undirected graph is called *directed*.
Moreover, a graph can be *weighted*, meaning that we attach a numerical value to every $e \in E$. In this case a graph can be seen as a triple $G = (V, E, W)$ where W is a function

$$W : E \to \mathbb{R}$$

Notice that a non weighted graph, sometimes called binary graph, can simple seen as a weighted graph where the weights are all 1. In general we expand W to a function over $V\mathrm{x}V$ where any $(k, l) \notin E$ has weight 0.
Generally, a graph can be represented by an **adjacency matrix** $A \in \mathbb{R}^{n,n}$ where $A_{ij} = W(i, j)$

Definition 4.2. A *ranking* or **centrality measure** S on a graph is any function assigning a numerical score to every node in V

$$S : G \to \mathbb{R}^n$$

Some classical examples of ranking on graphs are degree, eigenvector, PageRank and betweenness centrality, even though a wide variety is available. Broadly

speaking, the aim of centrality measures is to assign a higher score to nodes that are more important, or central, according to a given criterion.
Despite the large availability of ranking algorithms, apt to a variety of different scenarios and applications, relatively low attention has been brought to *dynamic scenarios* where, instead of a single graph G, we observe a sequence $(G_t)_{t=1,\dots,T}$ with, for our applications, a fixed vertex set $G_t = (V, E_t, W_t)$. It is natural to assume then, that also the nodes' ranks evolve in time, forming a sequence s_t.
To take into account the evolution of these ranks it is generally not enough to assume that $s_t = S(G_t)$. Consider for example a sports championship, where the time stamps are the single days where games are observed and the nodes, with relative ranks, the single teams. If we were to estimate a team's rank by just looking at single day, we could be offset by atypical events, that are not in line with the general trend of the previous matches observed, leading to peaky or incorrect rankings. Moreover, one could simply not be able to observe one or enough edges for every node at a given time t, being forced to appeal to the time dynamic for rank retrieval.

4.2 The probabilistic approch

Our proposed algorithm relies on a probabilistic approach and tries to smooth the rankings' changes using a time series connection. On a single time level, we rely on SpringRank [41], a novel ranking algorithm with a physically based principle. This choice is mainly motivated by the specific form of the algorithm, that allows a probabilistic interpretation and well adapts to our aims. In the following section we briefly introduce the static standard SpringRank, then moving to the dynamic version.

4.2.1 The static model

The SpringRank model has been presented in [41] as a physical metaphor, later turning into a pure optimization problem. Afterwards, the authors present a proof of how the solution can be interpreted also under a probabilistic bayesian view, which we summarize as follows

- **prior**

$$p(s) \propto \prod_{i \in V} \exp\left(-\frac{\alpha}{2}(s_i - 1)^2\right) \tag{4.1}$$

  product of uncorrelated normals

- **likelihood**

$$p(A|s) = \prod_{i,j} Pois(A_{ij}; \lambda_{ij} = ce^{-\frac{\beta}{2}H_{ij}}) \tag{4.2}$$

  where $H_{ij} := (s_i - s_j - 1)^2$

where, as from section 4.1, s are the rankings of the nodes in the network and A is the adjacency matrix. Notice that for now $\mathbb{R}^{n,n} \ni A = (A_{ij})_{i,j=1,\dots,n}$ and $\mathbb{R}^n \ni s = (s_k)_{k=1,\dots,n}$ are independent on time.
We make two remarks before presenting our dynamic version. First of all, notice that the likelihood has a mean dependent on the rankings through the value $H_{ij} = (s_i - s_j - 1)^2$. This specific shape was intended for two reasons in the original paper. First of all, the physical comparison was intended to reduce a total energy

$$E = \sum_{i,j} (s_i - s_j - 1)^2$$

In this case it would have been impossible to remove the -1 term, as an energy of type $\sum_{i,j}(s_i - s_j)^2$ is clearly minimized by setting all ranks equal. Secondly, the spring comparison was useful as an explanation of the fact that the bare *observation* of an edge between two nodes was, in the specific application, an hint towards the closeness of the two nodes' ranks. We will get rid of some of these assumptions in the next sections, giving detailed explanations for these modelling choices.

4.2.2 A first dynamic formulation

This dynamic version is an adaptation of the methods from [37] which, similarly, was trying to insert a time dependency in LDA (Latent Dirichlet Allocation [38], see section 3.4.3). This approach has also been adapted, for example, to recommender systems [39].
The main idea here is to introduce a time dynamic using an autoregressive process involving some latent variables. Formally, we define the model as follows:

- **prior**

$$\begin{aligned} p(s, \mu) &= p(s_0, \mu_0) \prod_{t=1}^{T} p(s_t, \mu_t | s_{t-1}, \mu_{t-1}) \\ &= p(s_0, \mu_0) \prod_{t=1}^{T} p(s_t|\mu_t) p(\mu_t|\mu_{t-1}) \end{aligned} \tag{4.3}$$

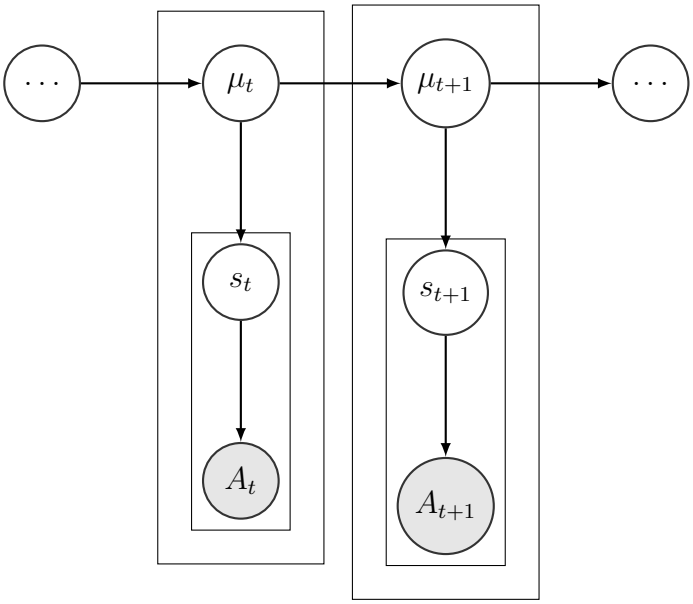

Figure 4.1: Graphical representation of the dynamic model

where the conditional distributions are:

$$s_t|\mu_t \sim N(\mu_t, \rho^2 \mathbb{1}) \tag{4.4}$$

$$\mu_{t+1}|\mu_t \sim N(\mu_t, \sigma^2 \mathbb{1}) \tag{4.5}$$

i.e. the underlying variables μ_t assume an autoregressive process form, and the s_t are generated like in the static model, but with means μ_t

- **likelihood**

$$p(A|s, \mu) = \prod_{t,i,j} Pois(A_{tij}; \lambda_{tij} = ce^{-\frac{\beta}{2} H_{tij}}) \tag{4.6}$$

where $H_{tij} := (s_{ti} - s_{tj})$

The generative model is represented in figure 4.1.

Notice that now all the terms have an extra index t, making the matrix of observations a tensor with dimensions T, N, N and the ranks a sequence of ranks, $s_t \in \mathbb{R}^n \, \forall t = 1, \ldots, T$.
As pointed out above, the autoregressive process is not on the ranks themselves, but rather on their means, allowing for another layer of stochasticity. Moreover, notice the modification of the likelihood with respect to the static form: here we consider a term $(s_{ti} - s_{tj})$. This is intended to actually remove the assumption that observing an edge between two nodes is a suggestion that they are close in

rank. In this way we can observe nodes with ranks at arbitrary distance. This is needed for the dataset under study in section 4.3. For an intuition of why this may be reasonable take for example two teams in the NBA league, where every teams plays against every other, or social networks, where a "follow" relationship can be shared between very important and very non central nodes.
Finally notice that the modelling flexibility of this method is very useful for a range of situations, and one should be able to pick the best assumptions and distributions that, in principle, allow the best fit to the data. In the next section we will instead turn to studying the inference for this model.

4.2.3 Choosing the method: generative model and inference

In the previous section we presented a possible form for the dynamic generative model, but many possible choices are available. For getting an idea on how to possibly tune one's choices in this context, as well as presenting two different types of inference, in the following sections we introduce some example variations on the main model proposed above. Finally, in section 4.2.4 we present the best performing model used for the experiments in section 4.3.

Inference for the dynamic model

The complicated form of the likelihood, together with the autoregressive form of the μ variables, doesn't allow for simple derivation of the posterior. Therefore we turn to using variational inference to retrieve an approximate posterior of the latent terms. In this section we present a possible choice for the inference. Here, the variational approximation q follows the ideas from [37] and falls under the structured mean field framework:

$$q(s, \mu) = \prod_{t,i} q(s_{ti}|\theta_{ti}, \eta_{ti}^2) \cdot \prod_i q(\mu_{:i}|\hat{\mu}_{:i}) \tag{4.7}$$

where $\mu_{:i} = (\mu_{1i} \dots \mu_{Ti}) \in \mathbb{R}^T$. The specific shape we use is $q(s_{ti}|\theta_{ti}, \eta_{ti}) \sim \mathcal{N}(\theta_{ti}, \eta_{ti}^2)$. For $q(\mu_{:i}|\hat{\mu}_{:i}) = \prod_t q(\mu_{ti}|\hat{\mu}_{:i})$ we consider a gaussian with mean and variance given by running a Kalman filter and smoother [42] on $\mu, \hat{\mu}$. Here $\hat{\nu}_t^2$ is an extra variational parameter that regulates the variance in the Kalman filter structure $\hat{\mu}_t|\mu_t \sim \mathcal{N}(\mu_t, \hat{\nu}_t^2 \mathbb{1})$.

Filtering m_t, V_t

$$
\begin{aligned}
m_t &\equiv \mathbb{E}(\mu_t|\hat{\mu}_{1:t}) = \left(\frac{\hat{\nu}_t^2}{V_{t-1}+\sigma^2+\hat{\nu}_t^2}\right) m_{t-1} + \left(1 - \frac{\hat{\nu}_t^2}{V_{t-1}+\sigma^2+\hat{\nu}_t^2}\right)\hat{\mu}_t \\
V_t &\equiv \mathbb{E}((\mu_t - m_t)^2|\hat{\mu}_{1:t}) = \left(\frac{\hat{\nu}_t^2}{V_{t-1}+\sigma^2+\hat{\nu}_t^2}\right)(V_{t-1}+\sigma^2)
\end{aligned} \tag{4.8}
$$

Smoothing $\tilde{m}_t, \tilde{V}_t$

$$
\begin{aligned}
\tilde{m}_{t-1} &\equiv \mathbb{E}(\mu_{t-1}|\hat{\mu}_{1:T}) = \left(\frac{\sigma^2}{V_{t-1}+\sigma^2}\right) m_{t-1} + \left(1 - \frac{\sigma^2}{V_{t-1}+\sigma^2}\right)\tilde{m}_t \\
\tilde{V}_{t-1} &\equiv \mathbb{E}((\mu_{t-1} - \tilde{m}_{t-1})^2|\hat{\mu}_{1:T}) = V_{t-1} + \left(\frac{V_{t-1}}{V_{t-1}+\sigma^2}\right)^2 (\tilde{V}_t - (V_{t-1}+\sigma^2))
\end{aligned} \tag{4.9}
$$

Using this model one can compute the ELBO and obtain

$$
\begin{aligned}
\mathbf{ELBO}(q) = &-\frac{1}{2\rho^2}\sum_{t,i}(\eta_{ti}^2 + \tilde{V}_{ti} + (\theta_{ti} - \tilde{m}_{ti})^2) - \frac{1}{2\sigma^2}\sum_{t,i}(\tilde{V}_{ti} + \tilde{V}_{t-1i} + (\tilde{m}_{ti} - \tilde{m}_{t-1,i})^2) \\
&+ \frac{\beta}{2}\sum_{t,i,j} A_{tij}(\theta_{ti} - \theta_{tj}) - c\sum_{t,i,j}\exp\left(\frac{\beta}{2}(\theta_{ti} - \theta_{tj}) + \frac{\beta^2}{8}(\eta_{ti}^2 + \eta_{tj}^2)\right) \\
&+ \frac{1}{2}\sum_{t,i}\log(2\pi e\eta_{ti}^2) + \log(2\pi e\tilde{V}_{ti}) \\
&- \frac{1}{2}TN\log(4\pi^2\sigma^2\rho^2) + \log c\sum_{t,i,j} A_{tij} + const
\end{aligned} \tag{4.10}
$$

extended computations are presented in section A of the appendix. Notice that this is one special case where the ELBO is actually available in closed form, making optimization easy via gradient descent techniques. Details on implementations will be presented in section 4.2.5.

While one may decide to stop here on the inference perspective, we actually decided to perform an extra optimization step. In fact, no action has been performed to infer the optimal hyperparameters of the model, such as β, c, σ^2 and ρ^2. As explained in section 3.4.1, we decided run a variational EM optimization loop, by alternating inference on q and optimization of the hyperparameters. Not only for the bayesian reasoning present in section 3.4.1, but also due to intermediate exploratory results, we just optimized with respect to the likelihood related terms β, c (when applicable, see next sections' models), leaving the prior's ones fixed.

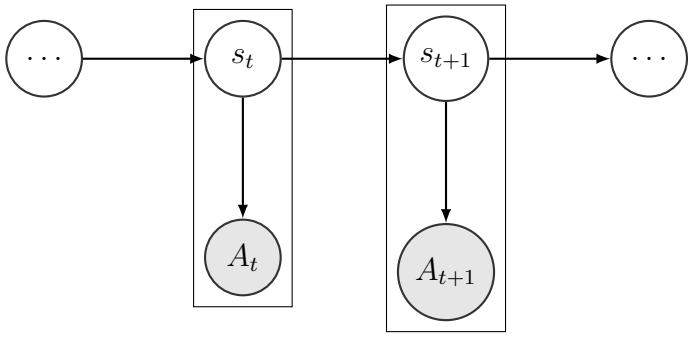

Figure 4.2: Graphical representation of the reduced form dynamic model

Playing with the likelihood: binomial distribution

In the adaptation from static to dynamic modelling, we tried to keep the common parts of the models unchanged. This includes a similar form for the prior of the ranks in 4.3 and the likelihood 4.6. However, upon knowledge of the number of observations $A_{tij} + A_{tji}$ at every time step t for a fixed pair i, j of nodes, the most appropriate likelihood to use would be binomial. In this case we would obtain

$$p(A|s\mu) = \prod_{t,i,j} Bin(p_{tij}, n_{tij}) \tag{4.11}$$

where $n_{tij} = n_{tji} = A_{tij} + A_{tji}$ and $p_{tij} = \frac{1}{1+e^{-2\beta(s_{ti}-s_{tj})}}$. Notice that choosing this form for p_{tij} we actually obtain $p_{tij} + p_{tji} = 1$ as it should be from the model. As we point out in the next paragraph, as well as in section 4.3, inference for this model is especially difficult, due to the use of a stochastic gradient affected by high variance. In order to reduce the number of parameters, as well as speeding calculations up, we refactor the model by removing a middle layer, see image 4.2.

Inference for the dynamic binomial model

We keep the same form for the variational approximation q as from the Poisson likelihood model 4.7, just taking into account the fact that we are dealing with a different number of variables, yielding

$$q(s|A) = \prod_{t,i} q(s_{:i}|\hat{s}_{:i})$$

where now the Kalman recursions are computed directly on the ranks s_{ti} with "variational observations" $\hat{s}_{ti}$. In this case the form of the ELBO is very similar.

In particular, the only different term to compute in

$$\mathbf{ELBO}(q) = \mathbb{E}_q[\log p(s)] + \mathbb{E}_q[\log p(A|s)] - \mathbb{E}_q[\log q(s)]$$

is the so called *reconstruction error* $\mathbb{E}_q[\log p(A|s)]$. It turns out, however, that this integral is impossible to express in closed form, due to the binomial likelihood shape, not allowing to write the ELBO explicitly. Despite this one only change then, the inference becomes a lot different. Here we are faced with two choices. Notice that the ELBO can be divided in three terms as above. In cases like ours, where two are explicitly computable, one may decide to optimize by computing the exact gradient over them, and find a stochastic approximation of the non computable term's gradient using the black box variational inference methods from section 3.2. On the other hand, one may simply consider an entirely stochastic estimate of the complete gradient, without using the explicit parts of the ELBO. During pilot trials we experimented with both methods.
We make one only remark here. While for the Poisson model there was no variance in the gradients used, and optimization proceeded smoothly, the use of stochastic estimates notably reduces the performances. This problem is made even worse by the use of a non mean field form for q, since we pass from the Kalman equations, that have the effect of entangling the variational parameters. This makes the usual variance reduction techniques less effective, as they depend on the number of parameters governing every part of the factorized q.
The likelihood (4.11) is invariant under multiplicative rescaling of β: by considering $\beta' = \lambda\beta$ for some $\lambda \in \mathbb{R}$, the same likelihood is obtained rescaling the ranks $s' = s/\lambda$. While the ELBO itself is not invariant under rescaling of β, we decided not to use a variational EM routine while optimizing this specific model, therefore fixing β in advance and optimizing via grid search together with the prior's parameters.

Mean field variational inference

While the structured approximation q presented in the previous sections can in principle yield good results, due to its similarity with the generative model, in practice one observes better results with a mean field formulation. While for brevity we don't present all the intermediate results for all the possible combinations of q approximation and inference choice, we try to motivate the final performances with a a posteriori analysis. As explained above, stochasticity in the gradient plays

an important role in the optimization and therefore in the final performances. Nonetheless, thanks to a likelihood that is supposedly closer to the real data, the binomial model yields improvements with respect to the Poisson likelihood one.
Similarly, one can play with the form of the variational approximation q. Before, we presented a structured choice that tries to restore a time dynamic introducing some "variational observations" to then estimate mean and variance of the ranks after the Kalman recursions. One could argue though, that after having found the first and second moment, the variables actually become independent and we end up with a form very similar to the mean field formulation. Moreover, with a mean field choice we can at least approximate the marginal moments of the ranks. Also, optimization is supposed to be cheaper, thanks to the reduced number of parameters and avoiding the passage through the Kalman filter and smoother, and thanks to the decreased variance of the gradient.
For all these reasons, and for the relative ease of implementation, we also experimented with a mean field approximation q, which is formulated as

$$q(s) = \prod_{t\,i} q(s_{ti}|m_i, V_i^2) = \prod_{t\,i} N(s_{ti}; m_i, V_i^2)$$

In the end, supposedly due to the reasons above, we find that this formulation yields the best performances for the binomial model, and therefore stick to this choice for the results presented in the next section.

4.2.4 The final model: choosing the likelihood

In the previous section we presented a detour of various choices that one could make while developing a generative model of this type, as well as two different possibilities for the variational approximation q. While we implemented and tested all of these, as well as other combinations of them, we present here what we found to be the best performing model, achieving results similar to the state of the art, as from section 4.3.
This model is a simple modification of the binomial formulation presented above, and starts from an intuition borrowed from [43]. A way of introducing a higher level of flexibility in the model, in fact, is to introduce the idea of *performances*. Ideally, when two players interact, the outcome can also be affected by some stochasticity independent of one's rank. For this reason we decide to make a small, but significant, change in the probability p_{tij} in the binomial likelihood 4.11 by choosing

$$p_{tij} = F(0; s_{ti} - s_{tj}, 2\,\beta^2) \tag{4.12}$$

where $F(x; \mu, \sigma^2)$ is the cdf of a gaussian distribution with mean μ and variance σ^2. This choice is justified by thinking about the outcome A_{tij} as due to some performances ξ_{ti}, ξ_{tj} distributed according to independent gaussians centered on the relative ranks

$$\xi_{ti} \sim \mathcal{N}(s_{ti}, \beta^2)$$

Therefore, one models the probability of getting p_{tij} as the probability of $\xi_{ti} - \xi_{tj} > 0$, i.e. one performance is higher than the other. Since $\xi_{ti} - \xi_{tj} \sim \mathcal{N}(s_{ti} - s_{tj}, 2\,\beta^2)$ one obtains formula 4.12.

Notice that we don't insert the performances directly into the formulation of the generative model, as they are just needed for obtaining the new p_{tij}. This means that they won't be estimated at inference time, and are not modelled by q. Explicit calculations for the posterior predictive distribution of $q(A|s)$ are included in appendix B. For the ELBO gradient calculation instead, one just needs to notice that the chain rule requires to compute, when using the reparametrization trick, the gradient of

$$\log p(A|s) = \sum_{t,i,j} A_{tij} \log p_{tij} + A_{tji} \log p_{tji}$$

with respect to s_{tij}. This is turn, requires to compute $\frac{\partial \log p_{tij}}{\partial s_{tij}} = \frac{\partial F(0; s_{ti} - s_{tj}, 2\,\beta^2)}{\partial s_{tij}}$, as well as $\frac{\partial \log p_{tij}}{\partial s_{tji}}$. While it is not straightforward to show, one is able to express the derivative of $F(0; \mu, \sigma^2)$ with respect to μ in closed form using the cdf and pdf of a gaussian, yielding closed form computations of the gradient. We refer to Autograd's implementation for the backpropagation of the gradient through the log-cdf [1]

4.2.5 Implementation details

All the models have been implemented in Python. Notice that, while the explicit form of the ELBO 4.10 and the iterative form of the Kalman filter and smoother allow for explicit (iterative) computations of the derivatives for the optimization, here we restored to the use of an automatic differenciation library. Among the many ones available for Python, for example Tensorflow [44], PyTorch [20], Caffe and Theano [45], we decided to use Autograd [46] for its convenient interface and easy integration with Numpy [47], which allows seamless implementation of the Kalman equations.

During the implementation of the model we were challenged with some issues. Among the notable ones there are the following:

[1] https://github.com/HIPS/autograd/blob/master/autograd/scipy/stats/norm.py

- some variables in the model are constrained to be strictly positive. While in principle some methods for constrained optimization are available, like projected gradient methods, to the best of our knowledge these haven't really been used in a variational inference framework. For our variables it's way easier to optimize some free variational parameters that are then cast, through a reversible and differenciable transformation, into the positive ones. For example when we needed to optimize the variational variances $\eta_{ti}^2 \in \mathbb{R}_{>0}$, we actually optimized some free variables $\tilde{\eta}_{ti} \in \mathbb{R}$ then using

$$\eta_{ti}^2 = f(\tilde{\eta}_{ti})$$

  where f is the *soft-plus* activation function

$$\begin{aligned} f : \mathbb{R} &\to \mathbb{R}_{>0} \\ x &\mapsto \log(1 + e^x) \end{aligned}$$

  this doesn't have any effect on the model, and requires only one extra step of automatic derivation via backpropagation. This trick has also been used for keeping β, c positive when performing the M-step during the variational EM optimization

- using the soft-plus activation is convenient on an optimization perspective, but comes with a caveat. In fact the function $\log(1 + e^x)$ is highly unstable on a numerical level, often leading to under or overflow impeding the convergence of the model. While for sums of type $\log(\sum_i e^{x_i})$ one usually resorts to the *log-sum-exp trick* [8], in our case we just had two summands, making the trick less effective. This specific instability is so well recognized that many programming languages have a specific function build for avoiding numerical errors. In our case we substituted the naïve implementation, which would impede convergence, with the function *logaddexp* included in (the Autograd wrapper of) Numpy. Indeed, this type of computations occurred often in the code, and we made extensive use of this optimized function, which turns out to be a staple of statistical/scientific programming

- as is the case for many Numpy dependent scenarios, the use of vectorized algebra sped the calculations up by a huge multiplicative factor. This is especially important due to the numerous multiplications and sums in the Kalman equations, which are not of trivial implementations in their tensor form

4.3 Experimental setting and results

4.3.1 From static to dynamic: related work

Before introducing natively dynamic algorithms, notice that it is relatively easy to turn any static ranking algorithm S into dynamic by just considering a rolling window of length k and $s_t = S(G_{t-k,t-k+1,\dots,t})$ where $G_{t-k,t-k+1,\dots,t}$ is a weighted graph with

$$W_{t-k,t-k+1,\dots,t} = \sum_{i=t-k}^{t} W_i$$

For example, if we stick to the teams and games example, the entry (i, j) in $G_{t-k,t-k+1,\dots,t}$ can be though as the sum of all the games won by j against i in the time window $[t-k, t]$. Notice, however, that this type of approach doesn't take into account a proper time dynamic, and relies on assuming a relative stability of the ranks in restricted time windows. Other drawbacks are given by the need of tuning the window length k, as well as an explicit rendering of the time dynamic, making interpretation difficult.
In the literature however, it is already possible to find attempts to model time evolving rankings. Notably, TrueSkill [48] and Whole History Rating (WHR) [49], both adopting a bayesian approach, albeit without recurring to any variational method. We therefore compare with these two algorithms in the following sections.

4.3.2 Evaluation method

We evaluate the ranks through their predictive ability, i.e. how accurate the forecasting of future games is given the most recent estimated ranks. In particular, given a training set until time t, we are provided with an estimated final rank s_t. We evaluate this ranks by assessing the predictions on the games for a certain time window $[t+1, t+m]$. Given the adjacency matrices $A_{t+1}, \dots, A_{t+m}$ we build a cumulative one A_{t+1}^{t+m} for the entire testing period, given by $\left(A_{t+1}^{t+m}\right)_{ij} := \sum_{\tau=t+1}^{t+m} A_{\tau ij}$. Moreover, while TrueSkill and WHR come provided with a way of extracting victory probabilities from the ranks, for VDR we find them as follows. Given a ranks vector s, typically the latest rank available s_t, and the relative variational means m_t and variances V_t^2, we compute the winning probability of i against j as

$$p_{ij} = \Phi\left(\frac{m_{ti} - m_{tj}}{\sqrt{2\beta^2 + V_{ti}^2 + V_{tj}^2}}\right)$$

where $\Phi(x)$ is the cdf of a standard gaussian computed in x. Explicit calculations can be found in appendix B.
The error measures that we use are the following:

- accuracy: fraction of correctly predicted outcomes. Precisely, if one has two different ranks s_{ti}, s_{tj}, a outcome $A = +1$ is correctly predicted if $s_{ti} > s_{tj}$. In case of integer values $A > 1$, we count the prediction as many times as the value of A is.

- log-likelihood: the value of the log-likelihood according to a Bernoulli distribution. Formally, we have

$$\text{log-lik} = \sum_{tij} A_{tij} \left(\delta_{A_{tij}>0} \log(p_{tij}) + \delta_{A_{tij}<0} \log(1 - p_{tij}) \right)$$

  where $\delta_{A_{tij}>0}$ is equal to 1 if $A_{tij} > 0$, else 0.

4.3.3 Experimental results

We evaluate our algorithms on the NBA dataset. The dataset consists of 218 aggregated time stamps and describing the games among 30 teams in the NBA league. In this context the adjacency matrix contains in A_{tij} the number of games won by i agains j during the time period t.
As from section 4.3.1, we compare against the TrueSkill and WHR algorithms, evaluating on accuracy and lok-likelihood of the models, as these are two classic measures for classification tasks. Notice that our interest still relies on retrieving reliable ranks, and this is only measured through the classification error measures, which are not the final goal.
All the models are trained on a moving window of length *train_ size* and are evaluated through their predictions on an adjacent testing window of length *test_ length*. For all algorithms we present the results obtained on the best training size found among 10, 20 and 40 periods, evaluating on different testing sizes. Regarding the best choice of the hyperparameters of VDR, we performed a grid search selecting the best performing model. Results are presented in table 4.1.

As we can see, our model is most of the time performing within one standard deviation from the alternative best one, outperforming the other. As a second comment, we can see that the standard error of our results is generally higher than the others'. This is an indication of the variability of the results due to the stochastic nature of the optimization. This is also an indication that, upon a more accurate selection of the best iteration attained during gradient descent, or using

	Optimal hyperparameters	test 1	test 5	test 10
TrueSkill	train: 40	acc: .644 ± 0.003 log-lik: 29.08 ± 9.66	acc: .667 ± 0.000 log-lik: 139.60 ± 10.91	acc: .646 ± 0.000 log-lik: 284.78 ± 13.43
WHR	train: 40	acc: .649 ± 0.005 log-lik: 27.70 ± 17.79	acc: .647 ± 0.001 log-lik: 139.38 ± 89.28	acc: .642 ± 0.001 log-lik: 281.63 ± 17.79
VDR binomial performance (ours)	train: 100 σ^2: 0.01 β^2: 1.0	acc: .646 ± 0.077 log-lik: 28.85 ± 5.45	acc: .643 ± 0.038 log-lik: 146.01 ± 13.32	acc: .643 ± 0.038 log-lik: 296.95 ± 19.78

Table 4.1: Results on NBA dataset. Given the optimal training length *train* and, for our algorithm, optimal σ^2 and β^2, we report accuracy and log-likelihood on different test sizes.

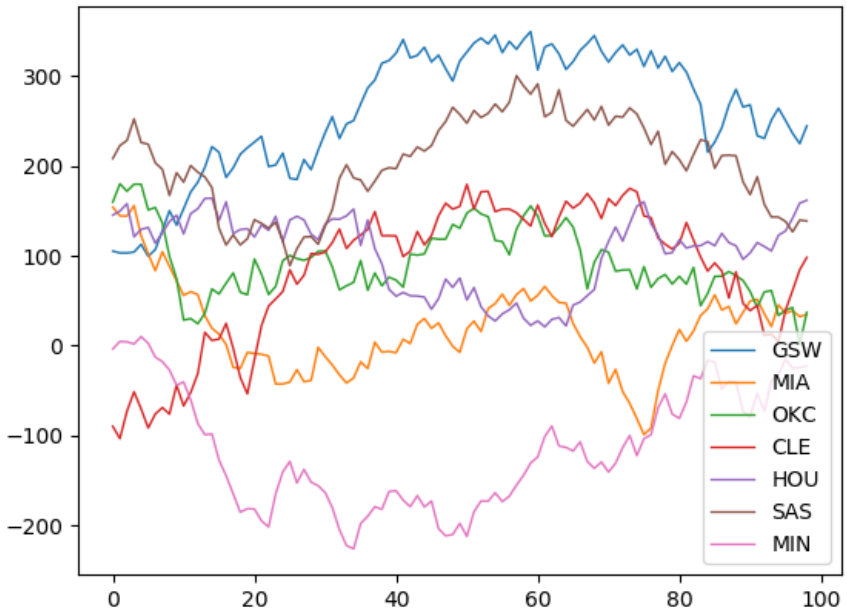

Figure 4.3: Posterior means of the ranks in time for some teams of the NBA dataset. The x axis indicates the time period, and the y axis the value of the estimated mean

alternative optimization methods, performances can be ameliorated. A plot of the estimated ranks in time for some of the teams in the dataset is presented in figure 4.3.

5. Future work and conclusions

In this thesis we gave a broad introduction to Variational Inference (VI), a method for posterior approximation in bayesian modelling. Specifically, we located VI in a broader inference context, and presented the latest innovations in the field, along with some of the major applications of this methodology.
Later, we introduced Variational Dynamic Ranking (VDR), a bayesian probabilistic model for dynamic ranking on graphs, and exploited these variational methods for performing inference in this context. Apart from our final chosen one, we decided to present some other options that we explored during the development of the algorithms, both on a modelling and on an inference perspective.
We showed that our method achieves state of the art comparable performances on different error measures.

Many lines of work are possible in this context. As we explained in the results section, scores seem to be affected by a great variability, due to the stochasticity of the gradient during the ELBO optimization. For the future we are willing to implement a more accurate optimization loop, taking into account the best score attained as opposed to the last iteration. We also propose the use of modern entropy based optimization techniques for the localization of better local optima, as well as natural gradient optimization routines. Another interesting extension is allowed by the exploitation of the generative nature of our model. In the future we will try to incorporate covariates used for the prediction of the results. External works suggest that this could bring to major improvements in the predictive capabilities, and in our case inference could be performed jointly for ranks and other covariates related parameters, allowing for better accuracy and log-loss.
On a more theoretical level, we will try to answer some open questions in the field, which is still actively investigated for more and more expressive variational approximations, such as kernel based distributions. Other interesting lines of research have recently opened due to the usage of statistical physics tools in variational inference. We will try to also move in this direction for future research.

Acknowledgements

A special thank goes to Caterina De Bacco for the support and availability shown throughout the development of the models, as well as for the trust and freedom allowed for my academic choices.
Other acknowledgements go, in no particular order, to: prof. Paolo Dai Pra, for the great effort taken towards making our master's degree as good as possible, way beyond what one could ask; Andrea Della Vecchia, for the collaboration and endless discussions during the development of our algorithms; my colleagues at Max Planck Institute, for making my life in and outside the office way better; my parents, for the support in all of the choices made, which have not been easy. Finally, thanks to every person involved in my journey until here. Because sometimes small things count more than one would think.

A. Appendix

A ELBO computations for section 4.2.2

We start from equations 3.2, 4.3, 4.6 and 4.7 to compute

$$\mathbf{ELBO}(q) = \mathbb{E}_q[\log p(s,\mu)] + \mathbb{E}_q[\log p(A|s,\mu)] - \mathbb{E}_q[\log q(s,\mu)] \tag{A.1}$$

and split this calculation in three parts

An useful lemma consider the random variable $s_{tij} := s_{ti} - s_{tj}$. Since for the variational distribution q the variables s_{ti} are uncorrelated normals, the variable s_{tij} also is, with

$$s_{tij} \sim N(\theta_{ti} - \theta_{tj}, \eta_{ti}^2 + \eta_{tj}^2) =: N(\theta_{tij}, \eta_{tij}^2)$$

Moreover, q is factorized over the s_{ti} so that

$$\begin{aligned}
\int (s_{ti} - s_{tj})^2 q(s) ds &= \int (s_{ti} - s_{tj})^2 q(s_{ti}, s_{tj}) ds_{ti} ds_{tj} \\
&= \int s_{tij}^2 q(s_{tij}) ds_{tij} \\
&= Var_q(s_{tij}) + \mathbb{E}_q[s_{tij}]^2 \\
&= \eta_{ti}^2 + \eta_{tj}^2 + (\theta_{ti} - \theta_{tj})^2
\end{aligned} \tag{A.2}$$

similar results hold for the random variables $s_{ti} - \mu_{ti}$ and $\mu_{ti} - \mu_{t-1,i}$, since all variables are independent for the variational distribution q.
Similarly, using the moment generating function of a gaussian $X \sim N(\mu, \sigma^2)$, that is

$$\mathbb{E}[e^{tX}] = \exp\left(\mu t + \frac{\sigma^2 t^2}{2}\right)$$

we obtain

$$\mathbb{E}_q[e^{\frac{\beta}{2} s_{tij}}] = \exp\left(\frac{\beta}{2}(\theta_{ti} - \theta_{tj}) + \frac{\beta^2}{8}(\eta_{ti}^2 + \eta_{tj}^2)\right) \tag{A.3}$$

First addend the first addend of A.1 is

$$\begin{aligned}
\mathbb{E}_q[\log p(s,\mu)] &= \int \sum_{t,i} \log[p(s_{ti}|\mu_{ti})p(\mu_{ti}|\mu_{t-1i})]q(s,\mu)dsd\mu \\
&= \int \sum_{t,i} \log[\frac{1}{\sqrt{2\pi\rho^2}} e^{\frac{-(s_{ti}-\mu_{ti})^2}{2\rho^2}}]q(s,\mu)dsd\mu \\
&\quad + \int \sum_{t,i} \log[\frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(\mu_{ti}-\mu_{t-1,i})^2}{2\sigma^2}}]q(\mu)d\mu \\
&= -\frac{1}{2\rho^2} \int \sum_{t,i} [(s_{ti}-\mu_{ti})^2]q(s,\mu)dsd\mu \\
&\quad - \frac{1}{2\sigma^2} \int \sum_{t,i} [(\mu_{ti}-\mu_{t-1,i})^2]q(\mu)d\mu - \frac{1}{2}TN\log(4\pi^2\sigma^2\rho^2) \\
&= -\frac{1}{2\rho^2} \sum_{t,i} (\eta_{ti}^2 + \tilde{V}_{ti} + (\theta_{ti}-\tilde{m}_{ti})^2) \\
&\quad - \frac{1}{2\sigma^2} \sum_{t,i} (\tilde{V}_{ti} + \tilde{V}_{t-1i} + (\tilde{m}_{ti}-\tilde{m}_{t-1,i})^2) - \frac{1}{2}TN\log(4\pi^2\sigma^2\rho^2)
\end{aligned} \tag{A.4}$$

Second addend using equations A.2, A.3 from the lemma above it is easy to compute the second addend of A.1:

$$\begin{aligned}
\mathbb{E}_q[\log p(A|s,\mu)] &= \mathbb{E}_q \left[\sum_{t,i,j} \log(\frac{\lambda_{tij}^{A_{tij}}}{A_{tij}!} e^{-\lambda_{tij}}) \right] \\
&= \sum_{t,i,j} \mathbb{E}_q \left[A_{tij}(\log c + \frac{\beta}{2} s_{tij}) - \log A_{tij}! - c e^{\frac{\beta}{2} s_{tij}} \right] \\
&= \sum_{t,i,j} \mathbb{E}_q \left[\frac{\beta A_{tij}}{2} s_{tij} - c e^{\frac{\beta}{2} s_{tij}} \right] + A_{tij}\log c + const \\
&= \sum_{t,i,j} A_{t,i,j}(\log c + \frac{\beta}{2}(\theta_{ti}-\theta_{tj})) - c\exp\left(\frac{\beta}{2}(\theta_{ti}-\theta_{tj}) + \frac{\beta^2}{8}(\eta_{ti}^2+\eta_{tj}^2) \right)
\end{aligned} \tag{A.5}$$

Third addend the third addend is just the entropy of a factorized gaussian. In fact in the variational distribution, given the "variational observations" $\hat{\mu}_{ti}$, the single latent variables μ_{ti} are independent gaussian with mean, variance $\tilde{\mu}_{ti}, \tilde{V}_{ti}$. Therefore from the formula for the entropy of a gaussian we obtain:

$$-\mathbb{E}_q[\log q(s,\mu)] = H(q) = \frac{1}{2} \sum_{t,i} \log(2\pi e \eta_{ti}^2) + \log(2\pi e \tilde{V}_{ti}) \tag{A.6}$$

B Predictive distribution for the final model in section 4.2.4

In this section we present the explicit calculations for the predictive distribution of the observations in section 4.2.4. While the posterior distribution of the ranks according to the mean field approximation q is given by $s_{ti} \sim \mathcal{N}(m_{ti}, V_{ti}^2)$, the observations depend on the ranks through the "performances" drawn from the ranks. In this section we show how to obtain the explicit distribution of the observations A given the posterior ranks s_{ti}.
We use the following lemma

Lemma A.1. Consider the cdf of a standard gaussian $\Phi(x)$. If $X \sim \mathcal{N}(\mu, \sigma^2)$ then

$$\mathbb{E}[\Phi(X)] = \Phi(\frac{\mu}{\sqrt{1+\sigma^2}}) \tag{A.7}$$

Proof. If Y is a standard normal by definition we have $\Phi(x) := \mathbb{P}(Y < x) \ \forall x \in \mathbb{R}$ Consider then that, if X and Y are independent random variables

$$\Phi(X) = \mathbb{P}(Y < X|X)$$

and therefore

$$\begin{aligned}
\mathbb{E}[\Phi(X)] &= \mathbb{E}[\mathbb{P}(Y < X|X)] \\
&= \mathbb{P}(Y < X) \\
&= \mathbb{P}(Y - X < 0) \\
&= \mathbb{P}\left(\frac{Y - X + \mu}{\sqrt{1+\sigma^2}} < \frac{\mu}{\sqrt{1+\sigma^2}}\right) \\
&= \Phi\left(\frac{\mu}{\sqrt{1+\sigma^2}}\right)
\end{aligned}$$

since $Y - X \sim \mathcal{N}(-\mu, 1+\sigma^2)$ □

To model the probability of an observation, we just need to compute the probabilities p_{tij}. As these are defined by the probability of the performance ξ_{ti} being

higher than ξ_{tj} we have

$$\begin{aligned}
p_{tij} &= \mathbb{E}_{s_{ti},s_{tj}\sim q}[\mathbb{P}(\xi_{ti} > \xi_{tj})] \\
&= \mathbb{E}_{s_{ti},s_{tj}\sim q}[\mathbb{P}(\xi_{tj} - \xi_{ti} < 0)] \\
&= \mathbb{E}_{s_{ti},s_{tj}\sim q}\left[\mathbb{P}\left(\frac{\xi_{tj} - \xi_{ti} - s_{tj} + s_{ti}}{\sqrt{2\,\beta^2}} < \frac{-s_{tj} + s_{ti}}{\sqrt{2\,\beta^2}}\right)\right] \\
&= \mathbb{E}_{s_{ti},s_{tj}\sim q}\left[\Phi\left(\frac{s_{ti} - s_{tj}}{\sqrt{2\,\beta^2}}\right)\right] \\
&= \mathbb{E}_{s_{tj}\sim q}\left[\mathbb{E}_{s_{ti}\sim q}\left[\Phi\left(\frac{s_{ti} - s_{tj}}{\sqrt{2\,\beta^2}}\right)|s_{ti}\right]\right] \\
&= \mathbb{E}_{s_{tj}\sim q}\left[\Phi\left(\frac{m_{ti} - s_{tj}}{\sqrt{2\,\beta^2 + V_{ti}^2}}\right)\right] \\
&= \Phi\left(\frac{m_{ti} - m_{tj}}{\sqrt{2\,\beta^2 + V_{ti}^2 + +V_{tj}^2}}\right)
\end{aligned} \tag{A.8}$$

where the last two passages are justified by the lemma above. As we can see the final prediction still depends on the difference between the posterior means, but the gaussian bell is wider due to the additional variance given by the usage of the performances, as well as from the posterior variances.

B. Bibliography

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

[2] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul, "An introduction to variational methods for graphical models," *Machine learning*, vol. 37, no. 2, pp. 183–233, 1999.

[3] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American Statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.

[4] F. J. Samaniego, *A comparison of the Bayesian and frequentist approaches to estimation*. Springer Science & Business Media, 2010.

[5] H. B. Stauffer, *Contemporary Bayesian and frequentist statistical research methods for natural resource scientists*. John Wiley & Sons, 2007.

[6] T. Hastie, R. Tibshirani, J. Friedman, and J. Franklin, "The elements of statistical learning: data mining, inference and prediction," *The Mathematical Intelligencer*, vol. 27, no. 2, pp. 83–85, 2005.

[7] M. Lavine, "Introduction to statistical thought," 2005.

[8] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[9] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[10] K. P. Murphy, "Conjugate bayesian analysis of the gaussian distribution," *def*, vol. 1, no. $2\sigma 2$, p. 16, 2007.

[11] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[12] M. Mezard, M. Mezard, and A. Montanari, *Information, physics, and computation*. Oxford University Press, 2009.

[13] D. Gamerman and H. F. Lopes, *Markov chain Monte Carlo: stochastic simulation for Bayesian inference*. Chapman and Hall/CRC, 2006.

[14] D. P. Kingma, "Variational inference & deep learning: A new synthesis," 2017.

[15] J. Drugowitsch, "Variational bayesian inference for linear and logistic regression," *arXiv preprint arXiv:1310.5438*, 2013.

[16] H. Robbins and S. Monro, "A stochastic approximation method," *The annals of mathematical statistics*, pp. 400–407, 1951.

[17] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2121–2159, 2011.

[18] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[19] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[20] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS-W*, 2017.

[21] M. Figurnov, S. Mohamed, and A. Mnih, "Implicit reparameterization gradients," in *Advances in Neural Information Processing Systems*, pp. 441–452, 2018.

[22] R. Ranganath, S. Gerrish, and D. Blei, "Black box variational inference," in *Artificial Intelligence and Statistics*, pp. 814–822, 2014.

[23] D. J. Rezende and S. Mohamed, "Variational inference with normalizing flows," *arXiv preprint arXiv:1505.05770*, 2015.

[24] E. G. Tabak, E. Vanden-Eijnden, *et al.*, "Density estimation by dual ascent of the log-likelihood," *Communications in Mathematical Sciences*, vol. 8, no. 1, pp. 217–233, 2010.

[25] E. G. Tabak and C. V. Turner, "A family of nonparametric density estimation algorithms," *Communications on Pure and Applied Mathematics*, vol. 66, no. 2, pp. 145–164, 2013.

[26] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling, "Improved variational inference with inverse autoregressive flow," in *Advances in neural information processing systems*, pp. 4743–4751, 2016.

[27] M. Gemici, D. J. Rezende, and S. Mohamed, "Normalizing flows on riemannian manifolds," *ArXiv*, vol. abs/1611.02304, 2016.

[28] R. Ranganath, D. Tran, and D. Blei, "Hierarchical variational models," in *International Conference on Machine Learning*, pp. 324–333, 2016.

[29] A. Birnbaum, "On the foundations of statistical inference," *Journal of the American Statistical Association*, vol. 57, no. 298, pp. 269–306, 1962.

[30] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," *arXiv preprint arXiv:1401.4082*, 2014.

[31] C. Doersch, "Tutorial on variational autoencoders," *arXiv preprint arXiv:1606.05908*, 2016.

[32] J. M. Tomczak and M. Welling, "Improving variational auto-encoders using householder flow," *arXiv preprint arXiv:1611.09630*, 2016.

[33] D. Kingma, T. Salimans, R. Josefowicz, X. Chen, I. Sutskever, M. Welling, *et al.*, "Improving variational autoencoders with inverse autoregressive flow," 2017.

[34] D. P. Kingma and P. Dhariwal, "Glow: Generative flow with invertible 1x1 convolutions," in *Advances in Neural Information Processing Systems*, pp. 10215–10224, 2018.

[35] P. K. Gopalan and D. M. Blei, "Efficient discovery of overlapping communities in massive networks," *Proceedings of the National Academy of Sciences*, vol. 110, no. 36, pp. 14534–14539, 2013.

[36] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley, "Stochastic variational inference," *The Journal of Machine Learning Research*, vol. 14, no. 1, pp. 1303–1347, 2013.

[37] D. M. Blei and J. D. Lafferty, "Dynamic topic models," in *Proceedings of the 23rd international conference on Machine learning*, pp. 113–120, ACM, 2006.

[38] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[39] L. Charlin, R. Ranganath, J. McInerney, and D. M. Blei, "Dynamic poisson factorization," in *Proceedings of the 9th ACM Conference on Recommender Systems*, pp. 155–162, ACM, 2015.

[40] S. M. Eslami, D. S. Tarlow, P. Kohli, and J. Winn, "Inference engine for efficient machine learning," Apr. 14 2016. US Patent App. 14/514,162.

[41] C. De Bacco, D. B. Larremore, and C. Moore, "A physical model for efficient ranking in networks," *Science Advances*, vol. 4, no. 7, 2018.

[42] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[43] P. Dangauthier, R. Herbrich, T. Minka, and T. Graepel, "Trueskill through time: Revisiting the history of chess," in *Advances in neural information processing systems*, pp. 337–344, 2008.

[44] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[45] Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions," *arXiv e-prints*, vol. abs/1605.02688, May 2016.

[46] D. Maclaurin, D. Duvenaud, and R. P. Adams, "Autograd: Effortless gradients in numpy," in *ICML 2015 AutoML Workshop*, vol. 238, 2015.

[47] T. Oliphant, "NumPy: A guide to NumPy." USA: Trelgol Publishing, 2006.

[48] R. Herbrich, T. Minka, and T. Graepel, "Trueskill$^{\text{TM}}$: A bayesian skill rating system," in *Advances in Neural Information Processing Systems 19* (B. Schölkopf, J. C. Platt, and T. Hoffman, eds.), pp. 569–576, MIT Press, 2007.

[49] R. Coulom, "Whole-history rating: A bayesian rating system for players of time-varying strength," in *Computers and Games*, 2008.