

# Variational optimization for the minimization of functions on the binary domain

Antti Ajanki

October 29, 2018

We consider the task of minimizing a non-linear, real-valued function of  $N$ -dimensional binary vector  $f: \{0, 1\}^N \rightarrow \mathbb{R}$ .

## 1 Variational optimization

The minimum of a function is always less than or equal to its expected value over any distribution  $p(x|\theta)$ :

$$\min f(x) \leq E[f(x)]_{p(x|\theta)}$$

The bound can be made tight if  $p(x|\theta)$  is so flexible that all the probability mass can be concentrated on the true minimum  $\operatorname{argmin} f(x)$ .

The idea of variational optimization [1] is to consider the upper bound  $U(\theta) = E[f(x)]_{p(x|\theta)}$  as a function of  $\theta$  and minimize that. This converts the task of minimizing a function  $f(x)$  of binary variables to minimization of a function  $U(\theta)$  of a continuous variable. Any method for continuous optimization can be applied for find a local minimum of  $U(\theta)$ .

## 2 Stochastic gradient descent

Because  $\mathbf{x}$  is a binary vector, it is natural to choose to take the expectation over a separable Bernoulli distribution:

$$p(\mathbf{x}|\boldsymbol{\theta}) = \prod_i p_i(x_i|\theta_i) = \prod_i \theta_i^{x_i} (1 - \theta_i)^{1-x_i}$$

Let's use the stochastic gradient descent to find the local minimum of the

upper bound  $U(\boldsymbol{\theta})$ . First we need the partial derivatives:

$$\begin{aligned}
\frac{\partial U(\boldsymbol{\theta})}{\partial \theta_j} &= \frac{\partial}{\partial \theta_j} E[f(\mathbf{x})]_{p(\mathbf{x}|\boldsymbol{\theta})} \\
&= \int f(\mathbf{x}) \frac{\partial}{\partial \theta_j} p(\mathbf{x}|\boldsymbol{\theta}) d\mathbf{x} \\
&= \int f(\mathbf{x}) \frac{\partial}{\partial \theta_j} \theta_j^{x_j} (1 - \theta_j)^{1-x_j} \prod_{i \neq j} p_i(x_i|\theta_i) d\mathbf{x} \\
&= \int f(\mathbf{x}) \left( \theta_j^{x_j} (x_j - 1)(1 - \theta_j)^{1-x_j-1} + x_j \theta_j^{x_j-1} (1 - \theta_j)^{1-x_j} \right) \prod_{i \neq j} p_i(x_i|\theta_i) d\mathbf{x} \\
&= \int f(\mathbf{x}) \theta_j^{x_j} (1 - \theta_j)^{1-x_j} \left( \frac{x_j - 1}{1 - \theta_j} + \frac{x_j}{\theta_j} \right) \prod_{i \neq j} p_i(x_i|\theta_i) d\mathbf{x} \\
&= \int f(\mathbf{x}) \left( \frac{x_j - 1}{1 - \theta_j} + \frac{x_j}{\theta_j} \right) \prod_i p(\mathbf{x}|\boldsymbol{\theta}) d\mathbf{x} \\
&= E \left[ f(\mathbf{x}) \left( \frac{x_j - 1}{1 - \theta_j} + \frac{x_j}{\theta_j} \right) \right]_{p(\mathbf{x}|\boldsymbol{\theta})}
\end{aligned}$$

David Barber proposed approximating the last expectation by sampling [2]:

$$\frac{\partial U(\boldsymbol{\theta})}{\partial \theta_j} \approx \frac{1}{K} \sum_{k=1}^K f(\mathbf{x}^{(k)}) \left( \frac{x_j^{(k)} - 1}{1 - \theta_j} + \frac{x_j^{(k)}}{\theta_j} \right),$$

where  $\mathbf{x}^{(1)}$  through  $\mathbf{x}^{(K)}$  are samples from  $p(\mathbf{x}|\boldsymbol{\theta})$ .

Now that we have a way to approximate the gradient  $\nabla U(\boldsymbol{\theta})$ , we can apply the stochastic gradient descent to iteratively search for the minimum. The  $\boldsymbol{\theta}$  is updated by taking small steps in the direction of the negative gradient:

$$\boldsymbol{\theta}^{\text{new}} = \boldsymbol{\theta} - \frac{\eta}{K} \sum_{k=1}^K f(\mathbf{x}^{(k)}) \left( \frac{x_j^{(k)} - 1}{1 - \theta_j} + \frac{x_j^{(k)}}{\theta_j} \right),$$

where  $\eta$  is the learning rate. Next, new  $\mathbf{x}$  samples are drawn from  $p(\mathbf{x}|\boldsymbol{\theta}^{\text{new}})$  and the iteration is repeated until convergence.

## References

- [1] Joe Staines, David Barber: *Variational Optimization*, <https://arxiv.org/abs/1212.4507v2>, 2012.
- [2] David Barber: *Evolutionary Optimization as a Variational Method*, <https://davidbarber.github.io/blog/2017/04/03/variational-optimisation/>, Apr 3, 2017.