# BOLTZIEMANMACHINE YO!

**Johan Nereng**

Department of Physics, University of Oslo, Norway

Spring, 2020

**Abstract**

her kjem det greier

***Author's comments:*** *Lalalla*

# 1 Introduction

This is an extention paper to my previous project paper *VMC: Effects of Importance Sampling and Jastrow factor on error estimates* [7], which is not published and not in review (not intended for publishing) https://github.com/johanere/FYS4411/tree/master/Project%201 . As such, this extention paper will not include descriptions of the theory already covered in [7].

**This project** introduces a new system, namely electrons in a harmonic oscillator, as such, the Hamiltonian of the system is changed from that of [7]. As the state of the system is represented by a neural network quantum state(NQS), the trial wave function is also new. These two alterations result in a different expression for the local energy, and parameter gradients. In addition, this project introduces a new VMC sampling method, Gibbs sampling.

As this is an extention paper, I will often reference the "orignail?" paaper, which in turns has relavnanat theory refkenmrken blalbal. USe some of the results from the orgina lpaper method and the "automated blocking" algorithm from [5].

Trene en RBM til å foreslå en god $\Psi$ slik at variational principle gir et godt estimat på $E_0$, det vil si lavest mulig - altså et tak.

In order to write this project paper and the code required to produce the results, I used a variety of tools, including: C++, Python 3.7.7, NumPy [8], as well as a number of books, web-pages and articles - of which most are listed under references. All the code required to reproduce the results may be found on my github page .

# 2 Material and methods

## 2.1 System: Electrons in 2D isotropic HO

The system consists of $P$ electrons in a $D$ dimensional isotropic harmoinc oscillator (HO) potential, with the following idealized total Hamiltonian, when using natural units, ($\hbar = c = e = m_e = 1$), and energies in atomic units a.u:

$$H = \sum_{i=1}^{P} \left( -\frac{1}{2}\nabla_i^2 + \frac{1}{2}\omega^2 r_i^2 \right) + \sum_{i=1}^{P} \sum_{j=1}^{i} \frac{1}{r_{ij}}, \tag{1}$$

Where $\omega$ is the oscillator frequency.

$$H_0 = \sum_{i=1}^{P} \left( -\frac{1}{2}\nabla_i^2 + \frac{1}{2}\omega^2 r_i^2 \right),$$

is the standard HO part of the Hamiltonian, while

$$H_1 = \sum_{i<j} \frac{1}{r_{ij}},$$

is the interactive part, where $r_{ij} = |\boldsymbol{r}_1 - \boldsymbol{r}_2|$, and $r_i = \sqrt{r_{i_x}^2 + r_{i_y}^2}$.

**The Pauli exlusion principle** requires that that two identical fermions, such as electrons, cannot occupy the same state [4]. As this project concerns itself with the ground state of a system of two non-interacting electrons, these electrons must therefore have opposite spins. This also means that the total spin of the two electron system in question must be zero, and that $P$ must be either one or two.

**Analytic solutions** for the system are available, which makes evaluating the performance of the algorithms possible under certain caveats. The energy of an harmonic oscillator is in one dimension is [4];

$$E_n = \hbar\omega(n + \frac{1}{2} \tag{2}$$

Thus, for a system of $P = \{1, 2\}$ non-interacting electrons in $D$ dimensions,

$$E_n = \hbar\omega(n + \frac{1}{2} \tag{3}$$

## 2.2 Neural network quantum state

Instead of seeking to find suitable trial wave function ansatz for the system, this project represents the state of the system by a neural network quantum state (NQS), as was done in by Carleo and Troyer [1]. In their article, they suggest that one may view the wave function as a computational black box, which for a specific system configuration, outputs an amplitude corresponding to the value of the wave function.

As was done in [1], I've chosen to use a Restricted Boltzmann machine [6][p.983] (RBM) when representing the state of the system. RBM models latent variables, or variables that are not directly observed, but rather inferred from other quantities - in this case, the local energy. The network consists of a layer of $M$ so called *visible nodes*, and a layer of $N$ *hidden nodes*. These layers have associated weights and biases, with a two-way feed forward. This enables the network both to output values for the hidden nodes as a function of the visible nodes, and vice versa. This means that an RBM is a generative model, in this case a model which can generate particle positions from a distribution. This distribution is a function of the weights and biases, as well as the hidden nodes.

**The marginal probability of the network** is given by

$$F_{rbm}(\boldsymbol{X}) = \frac{1}{Z}\sum_{h} e^{-E(\boldsymbol{X},\boldsymbol{h}},$$

and corresponds to the probability of the system state, $\boldsymbol{X}$. As $\boldsymbol{X}$ in this case is a vector of continuous particle positions, the energy function, $E(\boldsymbol{X}, \boldsymbol{h})$ of the network (not to be confused with the energy of the system), must be defined accordingly. There are various forms of RBMs with different energy functions, specialized to different types of data. The one used here, which enables continous $\boldsymbol{X}$, is known as a *Gaussian-Binary RBM* [6][p.986], and also includes binary hidden nodes $\boldsymbol{H} = \{h_j\}$ for $j = 1, 2, .., N$, and $h_j \in \{0, 1\}$. Using it's associated energy function, and representing each particle position as $X_i$, such that $i = 1, 2, .., M$ where $M = PD$, results in the following NQS/wave form representation [3];

$$\begin{aligned}\Psi(\mathbf{X}) &= F_{rbm}(\mathbf{X}) \\ &= \frac{1}{Z}\sum_{\{h_j\}} e^{-\sum_i^M \frac{(X_i-a_i)^2}{2\sigma^2}+\sum_j^N b_j h_j+\sum_{i,j}^{M,N}\frac{X_i w_{ij} h_j}{\sigma^2}} \\ &= \frac{1}{Z}e^{-\sum_i^M \frac{(X_i-a_i)^2}{2\sigma^2}}\prod_j^N(1+e^{v(j)})\end{aligned} \tag{4}$$

$$v(j) = b_j + \sum_{i=1}^{M}\frac{X_i W_{ij}}{\sigma^2} \tag{5}$$

Since all evaluations of the wave function is in the context of finding a ratio of probabilities; $\frac{\Psi_{rbm}^{new}}{\Psi_{rbm}^{old}}$, and all other quantities are derivatives of $\ln\Psi_{rbm}$, the normalization constant $Z$ is not relevant.

### 2.2.1 Local Energy and drift force

As described in project 1 [7][2.2], the results obtained through the VMC methods used in this project relies on the variational principle. As such, the local energy

$$E_L(\mathbf{r}) = \frac{1}{\Psi_T(\mathbf{r})}H\Psi_T(\mathbf{r}). \tag{6}$$

is required in order to carry out the Monte Carlo integration for the ground state energy. Using the trial wave function (4), and (6) an analytic expression for the local energy may be fond. This is derived in detail in Appendix 1: Analytic expression for the local energy, and will not be discussed further here. Based on the same appendix, the correponding drift force, needed for the importance sampling [7][2.2.2], is;

$$F_i = \frac{2\nabla\Psi_T}{\Psi_T} = 2\left[-\frac{(X_k-a_k)}{\sigma^2}+\sum_i^N\frac{w_{kj}}{\sigma^2}\frac{1}{1+e^{-v(\boldsymbol{X},j)}}\right] \tag{7}$$

## 2.3 Updating parameters

As the weights of the RBM are initialized randomly, the resulting energy is unlikely be the best approximation the algorithm is capable of. In [7], the aim was to find $\alpha$ s.t the energy was minimized. Here, the aim is instead to optimize the network. This is done by optimizing the weights $\boldsymbol{X}$, and biases $\boldsymbol{a}$ and $\boldsymbol{b}$, using the energy as a cost function. Although this optimization is over multiple parameters of different types, the strategy for each parameter $\beta_i$ of parameter type $\beta$, which can either be $\boldsymbol{a}$, $\boldsymbol{b}$, or $\boldsymbol{X}$, is the same as for $\alpha$ described in more detail in [7][2.2.3]. The update scheme for parameter $\beta$ is;

$$\boldsymbol{\beta}_{k+1} = \boldsymbol{\beta}_k - \eta_k \boldsymbol{g}(\boldsymbol{\beta}_k), \tag{8}$$

where $\eta$ is the learning rate. (8) requires the gradient of the local energy w.r.t the parameters in $\boldsymbol{\beta}$, $\boldsymbol{g}(\boldsymbol{\beta}) = \nabla_\beta\langle E_L\rangle$. This is given by

$$\nabla_\beta\langle E_L\rangle = 2\left(\langle\frac{\bar{\Psi}_\beta}{\Psi[\beta]}E_L[\beta]\rangle - \langle\frac{\bar{\Psi}_\beta}{\Psi[\beta]}\rangle\langle E_L[\beta]\rangle\right),$$

where $\frac{\bar{\Psi}_\beta}{\Psi[\beta]} = \frac{1}{\Psi_{rbm,\beta}}\frac{\partial\Psi_{rbm,\beta}}{\partial\beta}$. Expressions for $\beta = a, b, W$ can be found in Appendix 2: Derivatives w.r.t RBM parameters.

## 2.4 Gibbs Sampling

Gibbs sampling [6][ch.24.2] is a Markov Chain Monte Carlo algorithm, analogous to coordinate descent. In this implementation, the two-way feed forward network is used to sample the joint probability of $\boldsymbol{X}$ and bm $\boldsymbol{H}$ in a two step process. First, $P(\boldsymbol{H}|\boldsymbol{X})$ is used to determine the state of the binary nodes $H$, for the current system state $\boldsymbol{X}$. Then, after having updated $\boldsymbol{H}$, $P(\boldsymbol{X}|\boldsymbol{H})$ is used to sample a new system state. In contrast to the Metropolis Hastings algorithm, the probability of accepting this proposed system state equals one [2], ie. every proposal is accepted. The conditional probabilities are given by;

$$\begin{aligned} P(H_j = 1|\boldsymbol{X}) &= \frac{1}{1 + e^{-b_j - \sum_i^M \frac{X_i W_{ij}}{\sigma^2}}} = \text{logistic}(-(v(j)) \\ P(H_j = 0|\boldsymbol{X}) &= \text{logistic}((v(j)) \end{aligned} \tag{9}$$

and

$$P(X_i|\boldsymbol{H}) = \mathcal{N}(X_i; a_i + \boldsymbol{W}_{i*}\boldsymbol{H}, \sigma^2) \tag{10}$$

In order to sample $\Psi_{T,Gibbs}(\boldsymbol{X})$ s.t $|\Psi_{T,Gibbs}(\boldsymbol{X})|^2$ can be interpreted as a probability, $\Psi_{T,Gibbs}(\boldsymbol{X}) = \sqrt{F_{rbm}(\boldsymbol{X})}$, under the assumption that the wave function is positive definite. This slightly changes the expressions the local energy and parameter gradients, as is described briefly in Appendix 3: Local energy and parameter gradients for Gibbs sampling.

## 2.5 Algorithm

burn-in phase or equilibration phase [6][p.856]

**Algorithm: VMC using RBM with parameter optimization**

1. Initialize algorithm
   - Set number of iterations, $G$
   - Set the number of Monte Carlo cycles, $MC$, and "burn in fraction".
   - Set VMC method
     - Set expression for $E_L$ and $\frac{\bar{\Psi}_\beta}{\Psi[\beta]}$ according to choice of method
     - If using a variant of the Metropolis Algorithm, set step length, $l$
   - Initialize RBM parameters $\boldsymbol{a} = \mathcal{U}(0, 1/M), \boldsymbol{b} = \mathcal{U}(0, 1/N), \boldsymbol{W} = \mathcal{U}(0, 1/(M \times N)$
2. Execute Monte Carlo scheme
   - Initialize visible nodes, $\boldsymbol{X} = \mathcal{U}(-1, 1)$, set $E = 0$
   - For $cycle = 1, 2, .., MC$
     - Propose new configuration according to VMC method
     - Evaluate proposal, reject or accept.
     - If $cycle \geq MC\times$ "burn in fraction", calculate $E_L$ using current state.
3. Calculate $\langle H \rangle = \frac{E}{M}$
4. Optimize parameters $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{W}$
5. If the number of iterations of step 2 is less than $G$, go to step 2. Else, end simulation

### 2.6 Error analysis

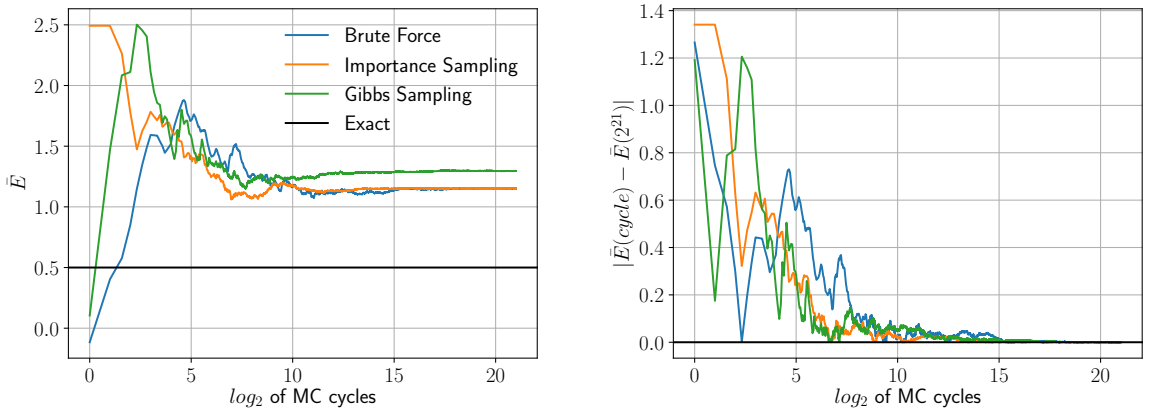
In [7] I found that the error estimate, $\sigma$, on correlated data without methods such as "blocking" is large deceptive. Hence, I have chosen to evaluate the error by the error estimate using "blocking", $\hat{\sigma}$. As I am already using $\sigma$ as the variance of the distribution, this is okey. $\hat{\sigma}$ is the error, and will not use any other type of error.

## 3 Results

### 3.1 Initial testing of VMC methods

In order to ascertain whether or not the implementation of the various VMC methods were successful, I ran an MC simulation over $2^{21}$ cycles $P = 1$, $D = 1$, $N = 2$, using Brute Force ($l = 1.0$), Importance Sampling ($\Delta t = 1.0$), and Gibbs Sampling.

Figure 1 shows the estimated energy $\bar{E}$ (left), and the absolute difference between the estimated energy at each cycle $\bar{E}(cycle)s$ and final estimated energy $\bar{E}$ (right), using the same seed for all three simulations. $\bar{E}$ is, as may be expected from an untrained network, relatively far from the exact energy. It is however clear that all three methods find a more-or-less stable value for $\bar{E}$ after a burn-in phase. As the blocking method requires data of a length which is a power of 2, I will henceforth discard all samples taken in the first half of the MC cycles (as I run cycles in powers of 2). This should (more than) ensure that samples in the burn-in phase is discarded, as well as provide a




(a) Energy estimate

(b) Difference in energy estimate from final estimate

Figure 1: Initial test of the three VMC methods (same seed)

suitable data format for the automated blocking method. Similarly stable $\bar{E}$ values after burn-in was observed in subsequent tests using different seeds. As such, the VMC methods are deemed successfully implemented.

Having established working implementation, I did a survey of the effects of $l$ (BF) and $\Delta t$ (IS). Appendix 4: Testing step length $l$ and $\Delta t$ 2 and 3

### 3.2 Training the network, n=2

Based on figure 4 and 5

## 4 Conclusions

## References

[1] Giuseppe Carleo and Matthias Troyer. Solving the quantum many-body problem with artificial neural networks. *Science (New York, N.Y.)*, 355(6325):602–606, 2017.

[2] Vilde Moe Flugsrud. Solving quantum mechanical problems with machine learning. Master's thesis, University of Oslo, 2018.

[3] FYS4411 Project 2: The restricted Boltzmann machine applied to the quantum many body problem. https://github.com/CompPhysics/ComputationalPhysics2/tree/gh-pages/doc/Projects/2020, 2020.

[4] David J Griffiths. Introduction to quantum mechanics, 1995.

[5] Marius Jonsson. Standard error estimation by an automated blocking method, 2018.

[6] Kevin P. Murphy. *Machine learning : a probabilistic perspective*. Adaptive computation and machine learning. MIT Press, Cambridge, 2012.

[7] Johan Nereng. Vmc: Effects of importance sampling and jastrow factor on error estimates. https://github.com/johanere/FYS4411/tree/master/Project%201, 2019.

[8] Travis E Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

# Appendices

## A Appendix 1.

### A.1 Analytic expression for the local energy

$$\begin{aligned} E_L &= \frac{1}{\Psi_{rbm}} \sum_{i=1}^{M} \left( -\frac{1}{2}\nabla_i^2 + \frac{1}{2}\omega^2 X_i^2 \right) \Psi_{rbm} + \frac{1}{\Psi_{rbm}} \sum_{i<j}^{P} \frac{1}{r_{ij}} \Psi_{rbm} \\ &= \frac{1}{\Psi_{rbm}} \sum_{i=1}^{M} \left( -\frac{1}{2}\nabla_i^2 \Psi_{rbm} \right) + \sum_{i=1}^{M} \frac{1}{2}\omega^2 X_i^2 + \sum_{i<j}^{P} \frac{1}{r_{ij}} \end{aligned} \tag{11}$$

And

$$\begin{aligned} \frac{1}{\Psi_{rbm}}\nabla^2\Psi_{rbm} &= \frac{1}{\Psi_{rbm}}\nabla(\Psi_{rbm}\frac{1}{\Psi_{rbm}}\nabla\frac{1}{\Psi_{rbm}}) \\ &= (\frac{1}{\Psi_{rbm}}\nabla\Psi_T)^2 + \nabla(\frac{1}{\Psi_{rbm}}\nabla\Psi_{rbm}) \\ &= (\nabla \ln \Psi_{rbm})^2 + \nabla^2 \ln \Psi_{rbm} \end{aligned} \tag{12}$$

So

$$E_L = -\frac{1}{2}\sum_{k=1}^{M} \left( \nabla \ln \Psi_{rbm} \right)^2 + \nabla^2 \ln \Psi_{rbm} \right) + \sum_{k=1}^{M} \frac{1}{2}\omega^2 X_i^2 + \sum_{i<j}^{P} \frac{1}{r_{ij}} \tag{13}$$

First derivative:

$$\begin{aligned} \frac{1}{\Psi_{rbm}}\nabla_k \Psi_{rbm} &= \nabla_k \ln \Psi_{rbm} \\ &= \nabla_k \big( \ln \frac{1}{Z} - \sum_i^M \frac{(X_i - a_i)^2}{2\sigma^2} + \sum_j^N \ln(1 + e^{b_j + \sum_i^M \frac{X_i w_{ij}}{\sigma^2}}) \big) \\ &= -\frac{(X_k - a_k)}{\sigma^2} + \sum_j^N w_{kj} \frac{\exp(b_j + \sum_i^M \frac{x_i w_{ij}}{\sigma^2})}{1 + e^{b_j + \sum_i^M \frac{X_i w_{ij}}{\sigma^2}}} \\ &= -\frac{(X_k - a_k)}{\sigma^2} + \sum_j^N \frac{w_{kj}}{\sigma^2} \frac{1}{1 + e^{-v(\boldsymbol{X}, j)}} \end{aligned} \tag{14}$$

Second derivative:

$$\begin{aligned}\nabla_k^2 \ln \Psi_{rbm} &= \nabla_k \big( -\frac{(X_k - a_k)}{\sigma^2} + \sum_j^N \frac{w_{kj}}{\sigma^2} \frac{1}{1 + e^{-b_j - \sum_i^M \frac{X_i w_{ij}}{\sigma^2}}} \big) \\ &= -\frac{1}{\sigma^2} + \sum_j^N \frac{w_{kj}^2}{\sigma^4} \frac{e^{-v(\boldsymbol{X},j)}}{(1 + e^{-v(\boldsymbol{X},j)})^2}\end{aligned} \tag{15}$$

or

$$\nabla \ln \Psi_{rbm} = -\frac{(X_k - a_k)}{\sigma^2} + \sum_j^N \frac{w_{kj}}{\sigma^2} \text{logistic}(-v(j)) \tag{16}$$

and

$$\nabla^2 \ln \Psi_{rbm} = -\frac{1}{\sigma^2} + \sum_j^N \frac{w_{kj}^2}{\sigma^4} \text{logistic}^2(-v(j)) \exp(-v(j)) \tag{17}$$

Thus to compute $E_L$, one may use (13), (16) and (17).

# B Appendix 2.

## B.1 Derivatives w.r.t RBM parameters

Where $\frac{\bar{\Psi}_\beta}{\Psi[\beta]} = \frac{1}{\Psi_T[\beta]} \frac{d\Psi[\beta]}{d\beta}$. For parameter $k$: $\frac{1}{\Psi_T} \frac{\partial \Psi_T}{\partial \beta_k} = \frac{\partial}{\partial \beta_k} \log \Psi_T$

for $\beta = \boldsymbol{a}$:

$$\frac{\partial}{\partial a_k} \log \Psi_T = \frac{X_k - a_k}{\sigma^2} \tag{18}$$

for $\beta = \boldsymbol{b}$:

$$\frac{\partial}{\partial b_k} \log \Psi_T = \frac{1}{1 + e^{-v(\boldsymbol{X},k)}} \tag{19}$$

for $\beta = \boldsymbol{W}$:

$$\frac{\partial}{\partial W_{kl}} \log \Psi_T = \frac{X_k e^{v(\boldsymbol{X},l)}}{(1 + e^{v(\boldsymbol{X},l)})\sigma^2} = \frac{X_k}{(1 + e^{-v(\boldsymbol{X},l)})\sigma^2} \tag{20}$$

# C Appendix 3.

## C.1 Appendix 3: Local energy and parameter gradients for Gibbs sampling

As the only parts of $\Psi_T$ that is traceable in $E_L$ for this system are the derivatives of $ln\Psi_T$,

$$\Psi_{T,Gibbs} = \sqrt{F_{RBM}(\boldsymbol{X})} = \sqrt{\Psi_{rbm}} \tag{21}$$

Means that the expression for $E_L$ remains the same as before, with the exceptions

$$\nabla \ln \Psi_{rbm} = \frac{1}{2} \nabla \ln \Psi_{rbm} \tag{22}$$

and

$$\nabla^2 \ln \Psi_{rbm} = \frac{1}{2} \nabla^2 \ln \Psi_{rbm} \tag{23}$$

And similarly for the parameter gradients, specifically for parameter number $k$ of $\boldsymbol{\beta}$;

$$\frac{1}{\Psi_{T,Gibbs}} \frac{\partial \Psi_{T,Gibbs}}{\partial \beta_k} = \frac{1}{2} \frac{\partial}{\partial \beta_k} \log \Psi_{rbm} \tag{24}$$

# D Appendix 4.

## D.1 Appendix 4: Testing step length $l$ and $\Delta t$

I averaged the standard error (using blocking) $\hat{\sigma}$ from 12 experiments. In each experiment, I used 8 MC simulations with parameter optimization (learning rate $\eta = 0.3$), with $2^{21}$ MC cycles with a 0.5 burn-in factor, for the shown values of $l$ and $\Delta t$ in figures 3 and 2. I found that $l = 1.0$ and $\Delta t = 2.0$ gave the best results among the tested values.

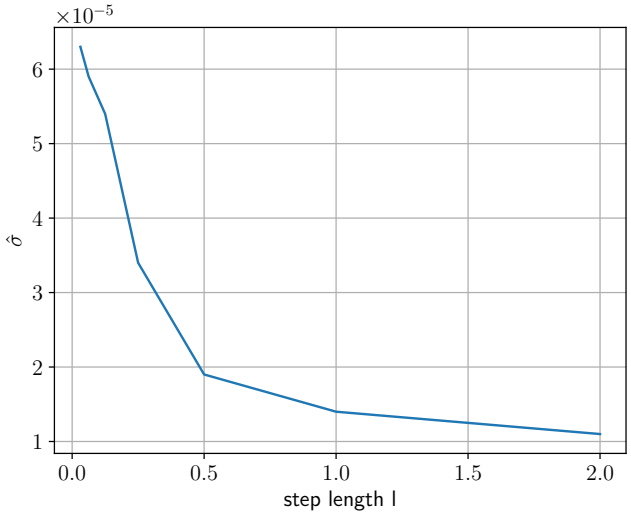



Figure 2: Importance Sampling: $\hat{\sigma}$ as a function of $\Delta t$

## D.2 Appendix 4: Testing learning rate $\eta$

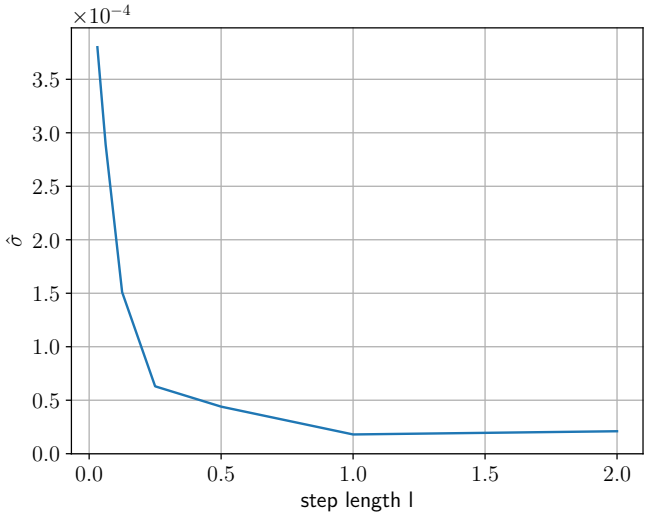



Figure 3: Brute Force $\hat{\sigma}$ as a function of $l$

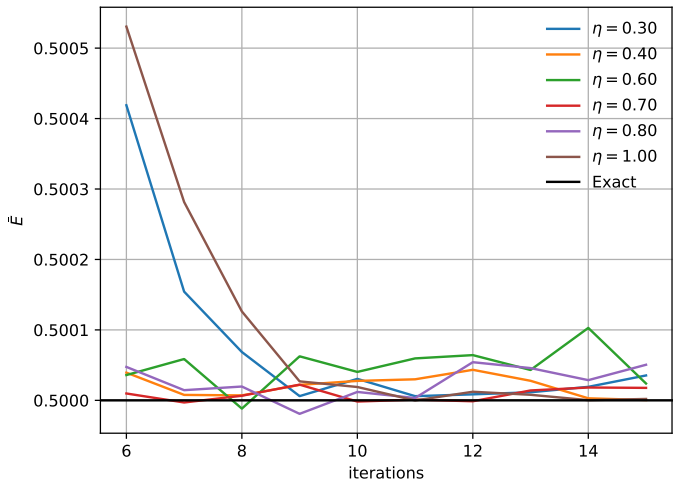



Figure 4: Brute Force: $\bar{E}$ for different values of $\eta$

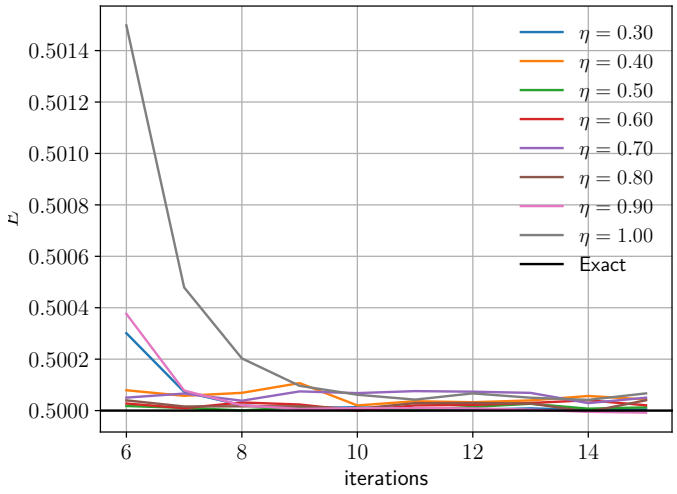



Figure 5: Importance Sampling: $\bar{E}$ for different values of $\eta$