# Report Week 3

Matthijs Neutelings, Dirren van Vlijmen, Olivier Brahma

February 10, 2022

## 1 Exercise 1

The first exercise requires the implementation of the Mean Field (MF) and the Belief Propagation (BP) approximation for the Ising model. The solution to this problem could be implemented in any preferred programming language, and the original files contain a rudimentary MATLAB implementation. This implementation includes the Ising model and the exact results in order to compare this with the MF and BP approximation.

### 1.1 Rewriting the code

In order to start the implementation of these models, we first rewrote the MATLAB file `\main_template.m` to a python file `\main_template.py`. The simple parameters such as `n=20;` and `w=J0/n+J/sqrt(n)*randn(n,n);` are translated with ease. There were 2 cases which required more complex solutions. Figure 1 shows these changes for `w` (weights within the sparse graph) and `sa` (all spin configurations).

### 1.2 MF implementation

The MF algorithm [1] starts with an initial randomised vector $m_0$ containing $n$ dimensions. The approximation uses the weights to update $m_i^{MF}$ and approach the exact mean values for $n$ spins ($m_i^{EX}$) (eq. 1).

$$m_i = \eta m_{i-1} + (1-\eta) tanh\Big(\sum_{j \in N(i)} w_{ij} m_{i-1j}\Big) \tag{1}$$

In order to reduce the computational costs $\epsilon$ is used to stop the algorithm if it has converged to a reasonable degree (fig. 3). When the difference between updated $m_i$ and $m_{i-1}$ is lower than $\epsilon$, the algorithm stops and the value for $m_i$ is compared to the exact solution (fig. 2). The figure shows that the convergence of the algorithm is prolonged for the $\eta$ values closer to 1.

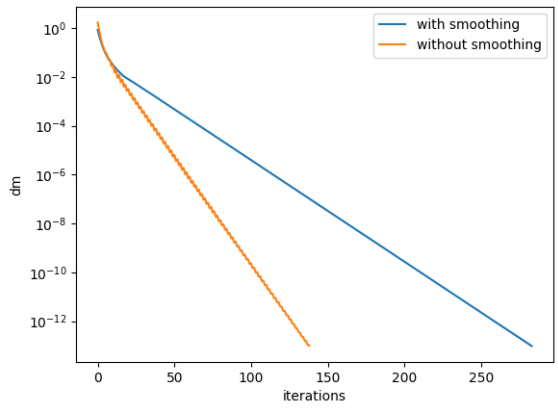
### 1.3 BP implementation

The BP algorithm implementation requires the calculation of the message passing equation (eq. 2). This represents the probability distribution over a single binary variable. Every iteration the $m_{ij}$ is used to update the matrix $a$, as it converges (eq. 3). Just as with the MF implementation, once the difference of $a$ ($da$) is lower than $\epsilon$, we save on computing costs by calling the algorithm converged (fig. 4). Finally, we added a maximum number of iterations. After 1500 iterations the BP algorithm stops as we found out that it sometimes does not seem to converge either due to our

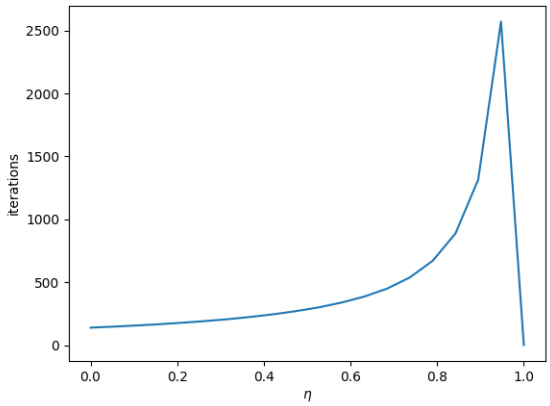
```
w = sprandsym(n,c1);

function ss=s_all(n)
    N=2^n;
    ss=2*(double(dec2bin(0:N-1))-48)-1;
end

sa = s_all(n); % all 2^n spin configurations
```

```
import scipy.stats as stats
import scipy.sparse as sparse

rvs = stats.norm().rvs
X = sparse.random(n, n, density=c1,
    data_rvs=rvs)
upper_X = sparse.triu(X)
result = upper_X + upper_X.T -
    sparse.diags(X.diagonal())
w = result.toarray()

sa = np.array(list(product([-1,1],
    repeat=n))) # all 2^n spin
    configurations
```

Figure 1: MATLAB file changes from to a python file. To our knowledge, this produces similar results[1].




(a) Difference in convergence for MF approximation with vs without smoothing on a fully connected graph with smoothing set to $\eta = 0.5$.




(b) The relation of the smoothing parameter $\eta$ on the number of iterations required for convergence.

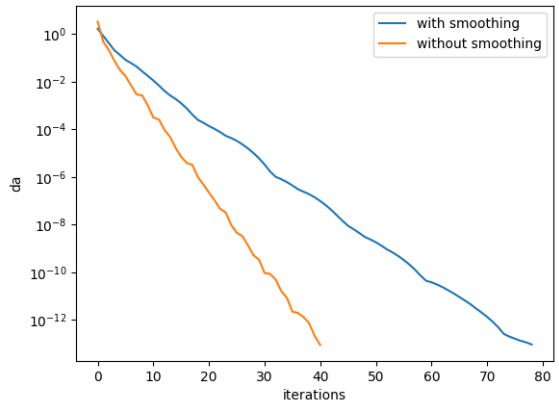
Figure 2: The impact the smoothing parameter has on the MF algorithm for a fully connected network with $\epsilon = 10^{-13}$, $J_0 = 0$, $J = 0.5$ and $J_{th} = 0.1$.

```python
def mf_approx(n, w, th, smoothing=0,
    eps=10**-13):
    m = np.random.normal(size=n) # random init

    dm = np.inf
    iterations = 0
    while(dm > eps):
        iterations += 1
        m_old = m
        m = smoothing * m + (1-smoothing) *
            np.tanh(np.dot(w,m) + th)
        dm = np.max(np.abs(m-m_old))

    return iterations, m
```

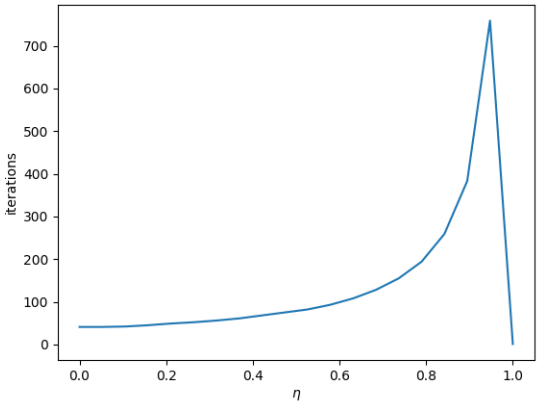
Figure 3: Python MF approximation algorithm with smoothing[1].

```python
def bp(n, w, th, c, smoothing=0, eps=10**-13):
    a = np.random.normal(size=(n,n)) # random
        init
    da = 1
    iterations = 0
    while(da > eps):
        iterations += 1
        a_old = a

        m_pos = 2*np.cosh(w + th +
            np.sum(np.multiply(a,c), axis=1) -
            np.multiply(a,c).T)
        m_neg = 2*np.cosh(-w + th +
            np.sum(np.multiply(a,c), axis=1) -
            np.multiply(a,c).T)

        a = .5 * (np.log(m_pos) - np.log(m_neg))
        a = smoothing*a_old + (1-smoothing)*a
        da = np.max(np.abs(a-a_old))

    m = np.tanh(np.sum(a, axis=1) + th)
    return iterations, m, a
```

Figure 4: Python BP approximation algorithm with smoothing[1].




(a) Difference in convergence for BP approximation with vs without smoothing on a fully connected graph with smoothing set to $\eta = 0.5$.




(b) The relation of the smoothing parameter $\eta$ on the number of iterations required for convergence.

Figure 5: The impact the smoothing parameter has on the BP approximation algorithm for a full connected network with $\epsilon = 10^{-13}$.

implementation or some other factor. Instead of running the algorithm again and hoping that it would converge at another instance, we decided to keep the authenticity by keeping these results.

$$m_{ij}(x_j) = 2cosh(w_{ij}x_j + \theta_i \sum_{k \in N(i) \setminus j} a_{ki}) \tag{2}$$

$$\begin{aligned} a_{ij}(x_j) &= \frac{1}{2} log \frac{m_{ij}(x_j = 1)}{m_{ij}(x_j = -1)} \\ &= \frac{1}{2} log \frac{2cosh(w_{ij} + \theta_i \sum_{k \in N(i) \setminus j} a_{ki})}{2cosh(-w_{ij} + \theta_i \sum_{k \in N(i) \setminus j} a_{ki})} \end{aligned} \tag{3}$$

Figure 5 shows the relation between the smoothing parameter and the number of iterations until convergence. Similar to the MF results (fig. 2) setting $\eta = 0.5$ reduces the convergence speed by about half. However, unlike the MF results, the BP convergence seems more volatile. The relation between $\eta$ and the number of iterations also matches the MF results. With the exception being the general number of iterations required until convergence.

# 2 Exercise 2

After implementing the MF and BP algorithms, we evaluated the performance of the approximations using the Root Mean Square Error (RMS), number of iterations until convergence, and the RMS error within the correlations. We will take a look at each of these respectively within the next subsections. For this exercise the algorithms are run on a fully connected network with $n = 20$ and 10 times for each iteration. This results in figure 6 where the mean and variance are plotted. These values are plotted against of $\beta$ for $0 < \beta \leq 2$. For the fully connected network with $J_0 = 0$ and $J_{th} = \beta$ the performances are measured as a function $\beta$. This value influences linear term $\theta \sim \mathcal{N}(0, \beta)$ and the quadratic term $w_{ij} \sim \mathcal{N}(0, \frac{\beta}{\sqrt{n}})$ (th and w resp. in fig. 3 and 4).

## 2.1 Accuracy

The RMS compares the $m_i^{MF}$ and $m_i^{BP}$ against the exact solution $m_i^{EX}$. We expect the RMS for MF to increase with $\beta$, since it should always converge, but it does not guarantee good error scores. While BP should yield good approximations [2]. The first thing that you might notice (as well as with the other plots) is that the values dip below zero. This is due to the variance being higher than the mean, resulting in these drops. The BP scores lower errors and is much more stable in general when compared to the MF approximation. This continues even for the higher $\beta$ values.

## 2.2 Iterations

For the number of iterations, we expected the number to be lower for the BP as it tries to find local approximations. While this was true for most variables, the BP algorithm did not always converge. Resulting in the hard stop at 1500

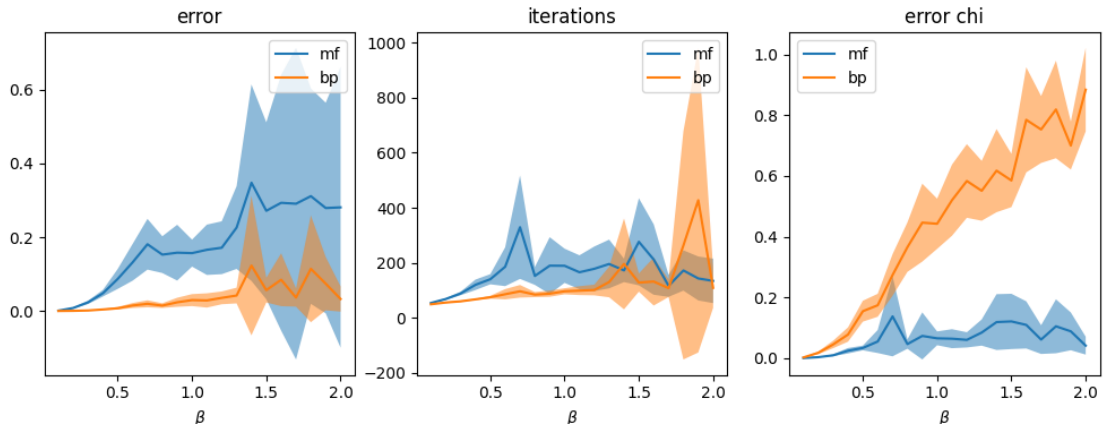



Figure 6: Average and variation of the accuracy of MF and BP approximation for fully connected model with $n = 20$ versus $\beta$. Left: RMS errors in mean $m_i$. Middle: number of iterations until convergence. Right: RMS error in connected correlations $\chi$. Smoothing $\eta = 0.5$.

iterations, which explains some of the spikes at around $\beta > 1.5$. The MF produces a stable number of iterations, without much influence from $\beta$, although generally higher.

### 2.3 Error in correlations

The RMS error in the correlations is calculated using equations 4 and 5 for MF and BP respectively.

$$\chi_{ij}^{MF} = \frac{\delta m_j}{\delta \theta_i} = A_{ij}^{-1} \tag{4}$$
$$A_{ij}^{-1} = \frac{\delta_{ij}}{1 - m_i^2} - w_{ij}$$

$$\chi_{ij}^{BP} = \sum_{x_i, x_j} x_i x_j b_{ij}(x_i, x_j) - m_i^{BP} m_j^{BP} \tag{5}$$
$$b_{ij}(x_i, x_j) = \frac{1}{Z} exp(w_{ij} x_i x_j = \theta_i x_i + \theta_j x_j + \sum_{k \in N(i) \setminus j} a_{ki} x_i \sum_{l \in N(j) \setminus i} a_{lj} x_j)$$
$$Z = \sum_{x_i, x_j} exp(w_{ij} x_i x_j = \theta_i x_i + \theta_j x_j + \sum_{k \in N(i) \setminus j} a_{ki} x_i \sum_{l \in N(j) \setminus i} a_{lj} x_j)$$

Using these correlations we can compute the error using equation 6 for both $\chi^{MF}$ and $\chi^{BP}$.

$$R_\chi = \sqrt{\frac{2}{N(N-1)} \sum_{i>j} (\chi_{ij} - \chi_{ij}^{EX})^2} \tag{6}$$

This indicates the error in the correlations of the approximations and the exact solution. The plot (fig. 6) shows that while the error of BP remains low and stable, the correlation error rises rapidly with $\beta$. The opposite happens with MF, where the correlation error is low and the error becomes higher and volatile.

## 3 Exercise 3

We have now seen how these approximations perform on fully connected networks. For this exercise, we will have to evaluate the performance on a partially connected network c $0 < c < 1$. The exercise asks for the $0 < c \leq 1$, which includes $c = 1$ (a fully connected network).

In order to implement this, we simply looped the program over variables $c$ and $\beta$ for different $\theta \in [0.1, 0.6, 1]$ [1]. When plotting the error/iterations over $c$ and $\beta$ we use the values $[0.1, 0.6, 1]$ for $\beta$ and $c$ respectively. The limited range of these values is in order to save on computing costs. The code and results are available online if required. The mean variance of the results (fig. 7, 8) are calculated on $n = 20$ and 6 runs for each setting.

### 3.1 Partial connectivity

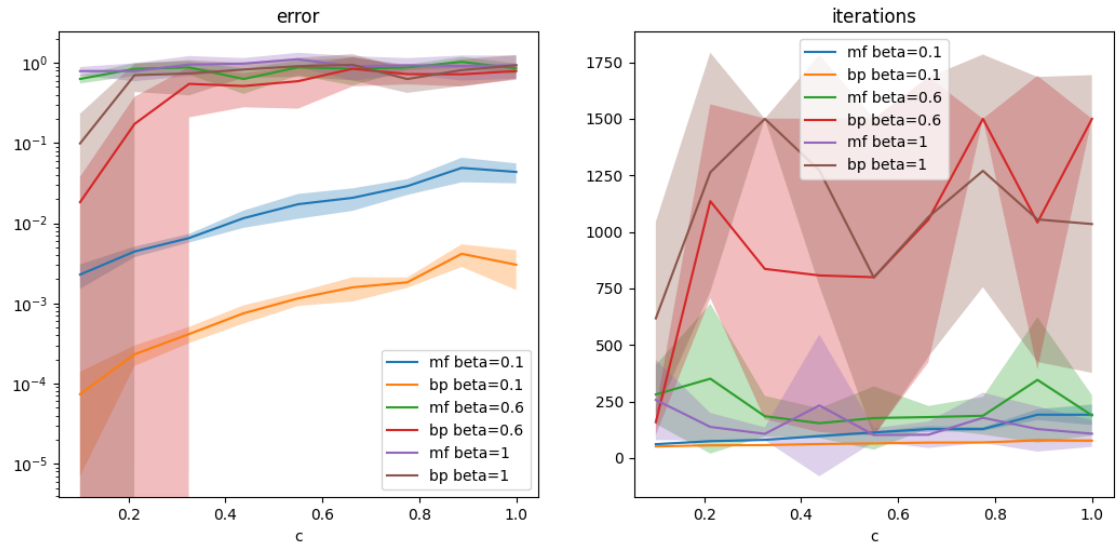
For the connectivity (fig. 7) we see a general increase with the error. Note that this is on a logarithmic scale. The performance also become much less stable with the increase of $\theta$, indicated by the variance increase. However, the performance mean values do seem to get better for both. The connectivity does not seem to impact the variance for either the BP or MF algorithm. Finally, the $\beta$ value seems to decrease error performance with the $\beta = 0.1$ scoring better in all connectivity values than the $\beta = [0.6, 1.0]$ values. BP seems to outperform MF in the error.
The number of iterations reveals something else. The impact of the two higher $\beta$ values is clearly visible. With the iteration ceiling of 1500 being reached multiple times for most values. Around $c = 0.5$ there does seem to be a dip for each $\theta$, however, this is not enough to draw conclusions from with only 6 iterations. The MF always converges, which shows that iterations are much lower than for BP. For $\beta = 0.1$ the performance influence of connectivity, $c$ is not noticeable with a stable mean and variance.

### 3.2 Partial $\beta$

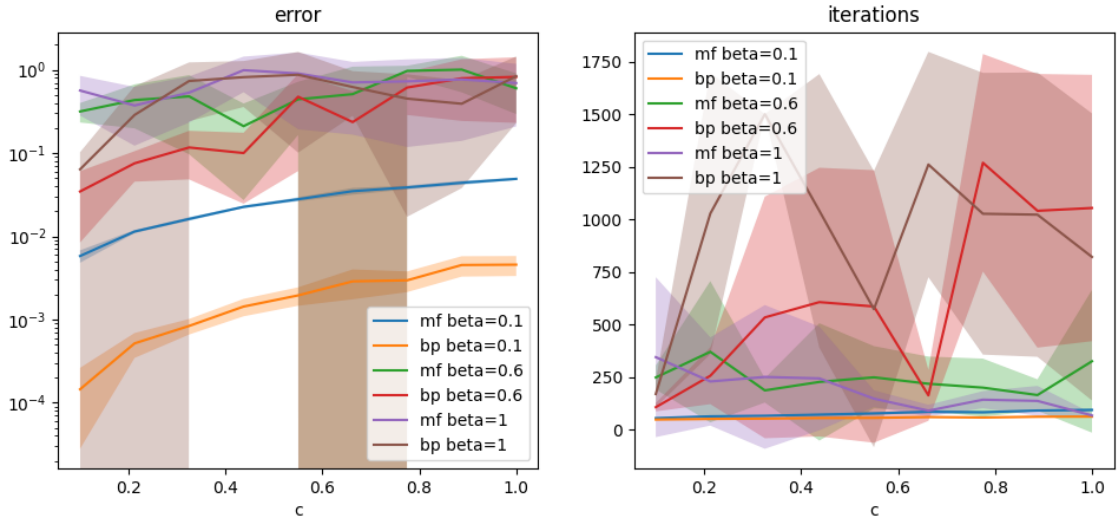
Just as with the connectivity, the $\beta$ plots show that the $\beta$ has much more influence on the performance than the connectivity has. Again, this performance also decreases with the increase of $\beta$. The influence the connectivity has on these plots is better defined for the higher $\theta$ and lower $\beta$ values. With the error starting off low and increasing with $\beta$. $\theta$ seems to have a positive correlation for every configuration of $c$ and $\beta$.
The number of iterations reveals little on the relation with $\beta$. One thing that happens is for $\beta = 1$ MF struggles to converge within 1000 iterations and since there is no limit to this, in some instances it even rises above 10000 iterations. Connectivity has little influence on the number of iterations and any conclusions drawn for this requires more testing. $\theta$ decreases the number of required iterations until convergence, reducing both the mean and variance for most values.

## 4 Conclusion

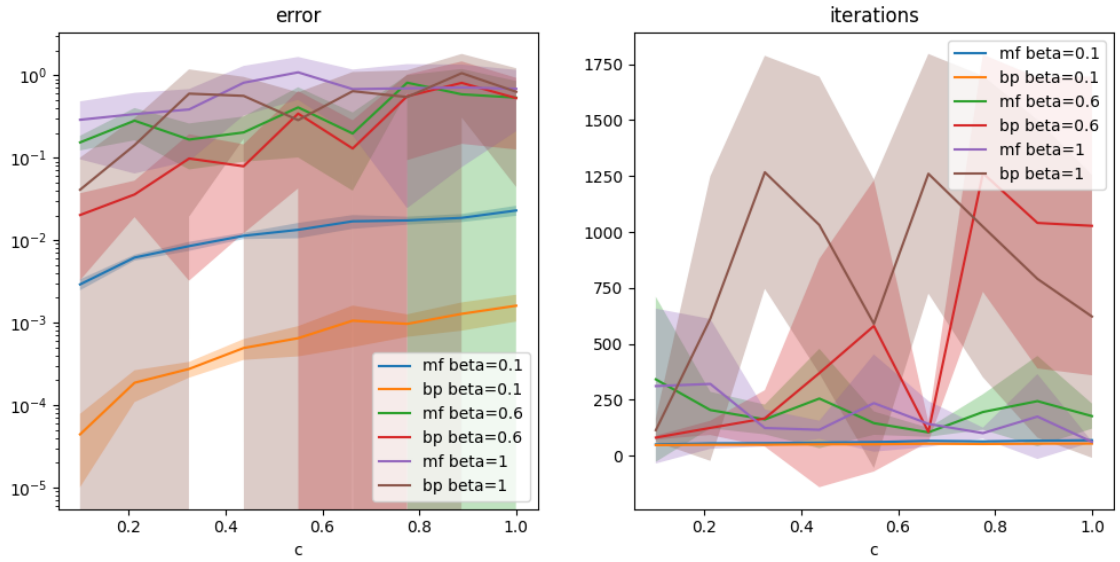
The implementation of the MF and BP approximation algorithms did not go very smooth. We struggled a lot and spend much more time than anticipated. Comparing BP and MF for a fully connected network shows that BP has better error scores, but suffers for the correlation error. While, MF performs better for the latter, but is much more volatile for the former. For a partially connected network, the exercise investigates not a single ($\beta$), but three parameters ($\beta$, $c$ and $\theta$). This revealed that the influence of $\beta$ is much higher than that of connectivity $c$, but both have a negative impact on the approximation error. $\theta$ seemed to increase the variance but decrease the error. The variables do not seem to impact the number of iterations for either algorithm all that much. Apart from the $\beta = 1$, where more configurations show a spike. Overall, the error for BP is lower, while the number of iterations required for convergence is better with the MF approximation.




(a) $\theta = 0.1$

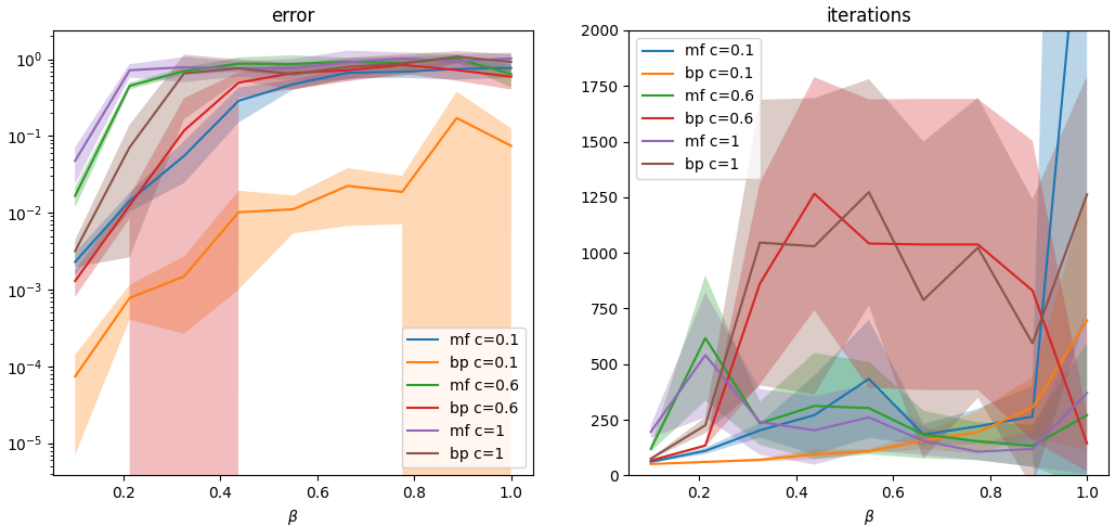



(b) $\theta = 0.6$

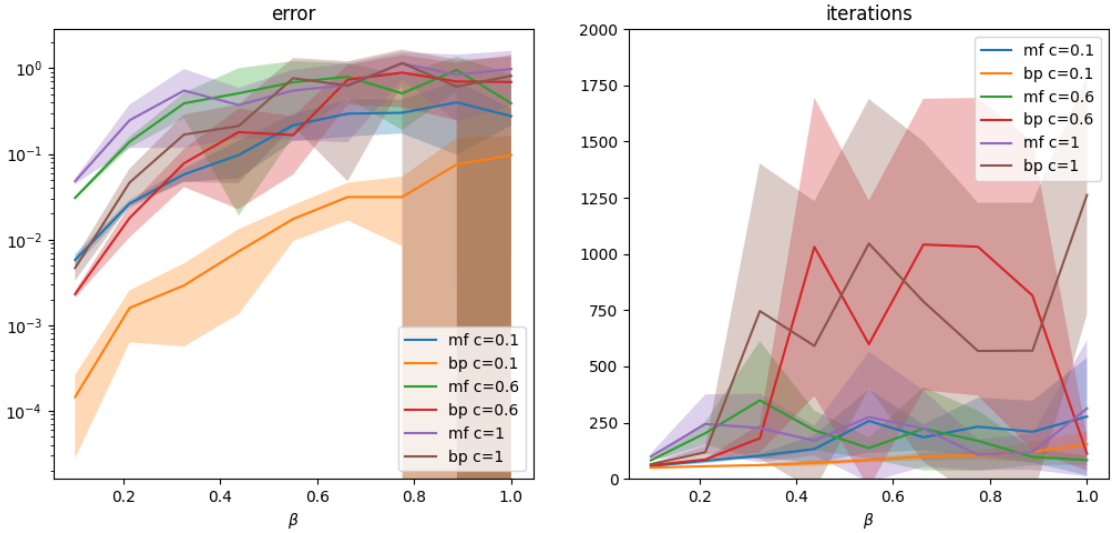



(c) $\theta = 1$

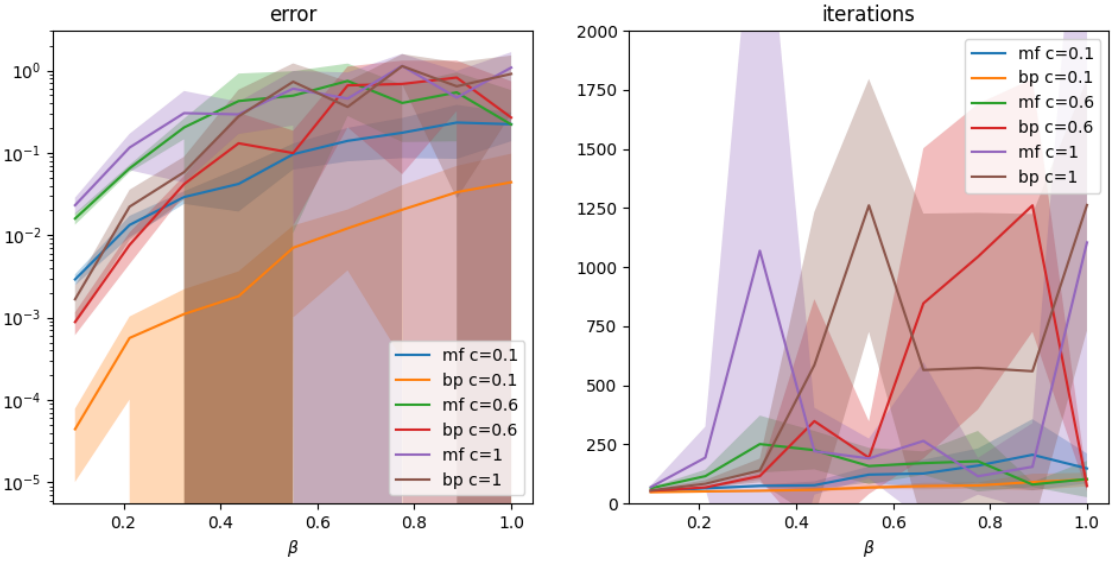
Figure 7: Errors and number of iterations for MF and BP on sparse model versus connectivity c. $n = 20$, and smoothing for both MF and BP is $\eta = 0.5$




(a) $\theta = 0.1$




(b) $\theta = 0.6$




(c) $\theta = 1$

Figure 8: Errors and number of iterations for MF and BP on sparse model versus $\beta$. $n = 20$, and smoothing for both MF and BP is $\eta = 0.5$

# References

[1]M. Neutelings, D. van Vlijmen, and O. Brahma, *Advanced-machine-learning: week 5*, https://github.com/Dirrenvv99/Advanced-Machine-Learning/tree/main/Week%205.

[2]E. Riegler, G. E. Kirkelund, C. N. Manchón, and B. H. Fleury, "Merging belief propagation and the mean field approximation: a free energy approach", in 2010 6th international symposium on turbo codes iterative information processing (2010), pp. 256–260.