University of Bucharest

Artificial Intelligence

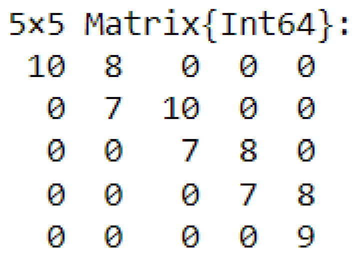
Recommendation system using Hierarchical Poisson Factorization

Student: Radu Madalin-Cristian

Group: 407

1) The dataset

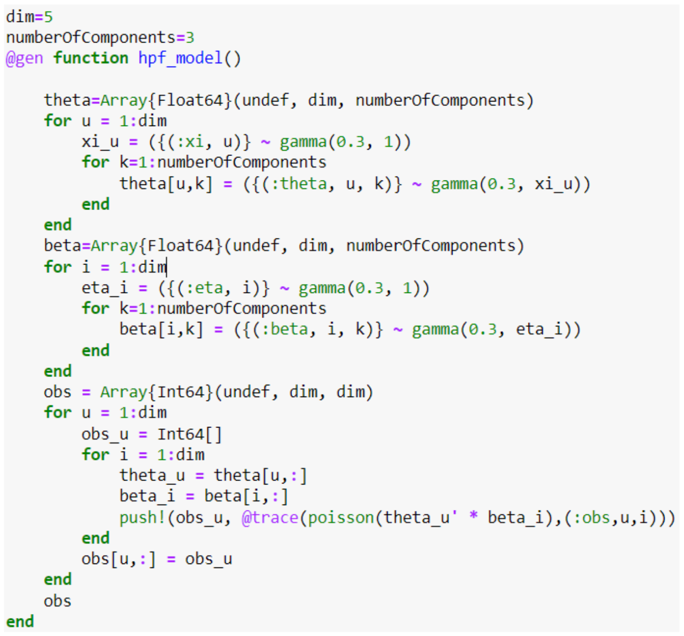
For this project I’ve chosen a book recommendation dataset taken from Kaggle [1]. The dataset contains general information about the books and the users who have rated them, but we are only interested in the ratings they’ve given. The dataset contains 1149780 ratings given. MCMC is pretty costly, so I will use 5 users and 5 items as a start in training the model. I’ve chosen the users and items by picking the first user randomly, and then, again randomly, picking a book that user has read. Then, for the rest of the books and users I chose a user that has read the last book chosen, who also read a greater amount of other books and then I repeated the process until I had 5 books and 5 users. The reason why I picked users who read a lot of books is because they have a higher chance of rating a book fairly. The rest of the picking process is pretty random, giving the data a sense of connection and a sense of randomness at the same time. With the given data, I created a 5 by 5 matrix which I filled with the ratings given to the books by the users. For the items that the users haven’t rated, I left a value of 0 in the matrix. The resulting matrix:



```
5×5 Matrix{Int64}:
 10  8   0  0  0
  0  7  10  0  0
  0  0   7  8  0
  0  0   0  7  8
  0  0   0  0  9
```

2) The implementation

Having the observed data in place, the next step is to create the hierarchal Poisson factorization model [2]:

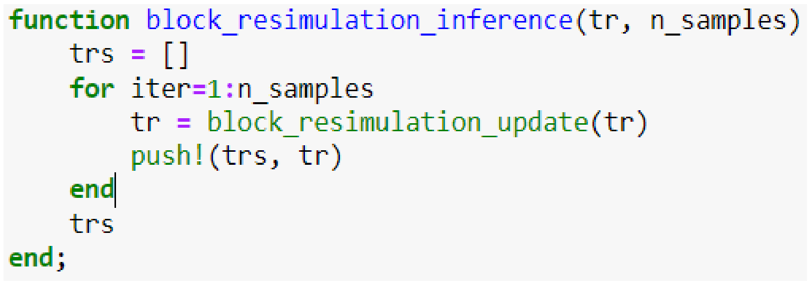


```
dim=5
numberOfComponents=3
@gen function hpf_model()

    theta=Array{Float64}(undef, dim, numberOfComponents)
    for u = 1:dim
        xi_u = ({(:xi, u)} ~ gamma(0.3, 1))
        for k=1:numberOfComponents
            theta[u,k] = ({(:theta, u, k)} ~ gamma(0.3, xi_u))
        end
    end
    beta=Array{Float64}(undef, dim, numberOfComponents)
    for i = 1:dim
        eta_i = ({(:eta, i)} ~ gamma(0.3, 1))
        for k=1:numberOfComponents
            beta[i,k] = ({(:beta, i, k)} ~ gamma(0.3, eta_i))
        end
    end
    obs = Array{Int64}(undef, dim, dim)
    for u = 1:dim
        obs_u = Int64[]
        for i = 1:dim
            theta_u = theta[u,:]
            beta_i = beta[i,:]
            push!(obs_u, @trace(poisson(theta_u' * beta_i),(:obs,u,i)))
        end
        obs[u,:] = obs_u
    end
    obs
end
```

For the number of preferences and attributes I chose a default value of 3 so as to not overload the model when training.

The next step is to create an initial trace which encapsulates the observed data and the initial values of the latent variables. After that, all is left for training the model is to make the inference function:



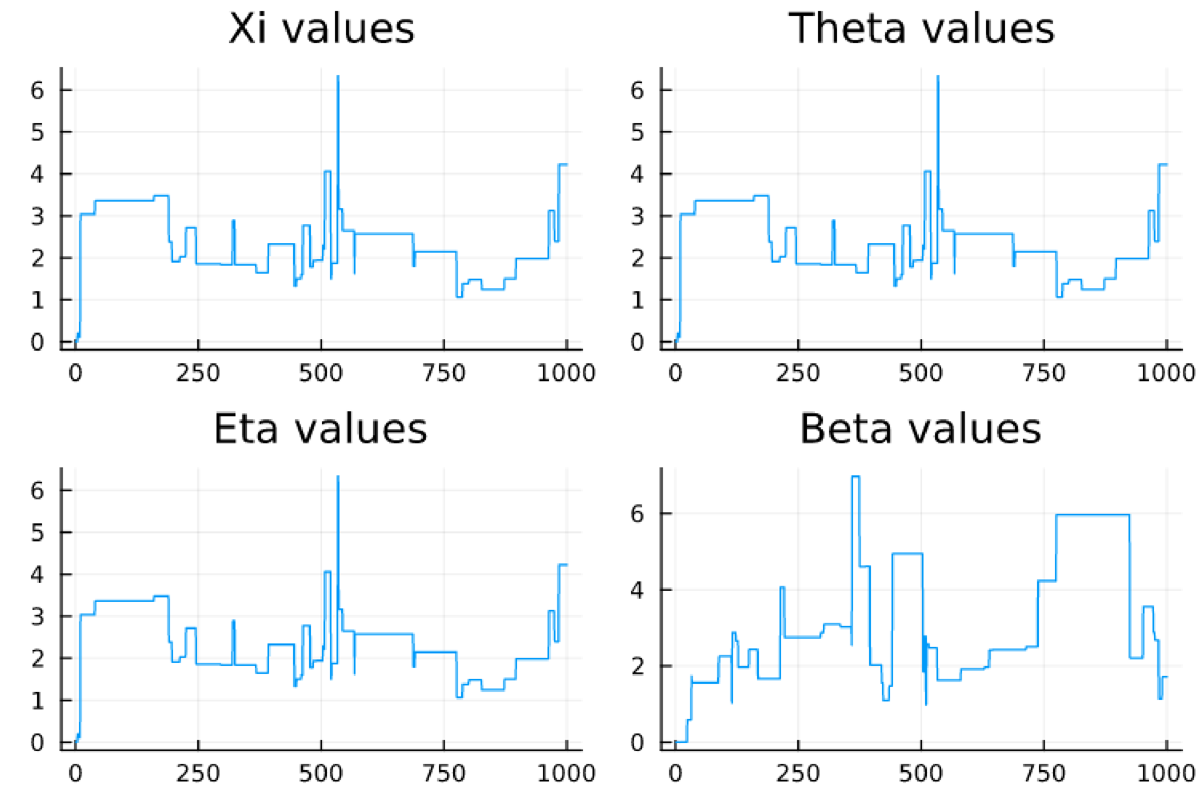
```
function block_resimulation_inference(tr, n_samples)
    trs = []
    for iter=1:n_samples
        tr = block_resimulation_update(tr)
        push!(trs, tr)
    end
    trs
end;
```

This uses the block_resimulation_update function which updates the latent variables for each user and item using the Metropolis-Hastings algorithm:

```julia
function block_resimulation_update(tr)
    for u=1:dim
        latent_variable = Gen.select((:xi,u))
        (tr, _) = mh(tr, latent_variable)
        for k=1:numberOfComponents
            latent_variable = Gen.select((:theta,u,k))
            (tr, _) = mh(tr, latent_variable)
        end
    end
    for i=1:dim
        latent_variable = Gen.select((:eta,i))
        (tr, _) = mh(tr, latent_variable)
        for k=1:numberOfComponents
            latent_variable = Gen.select((:beta,i,k))
            (tr, _) = mh(tr, latent_variable)
        end
    end
    tr
end
```

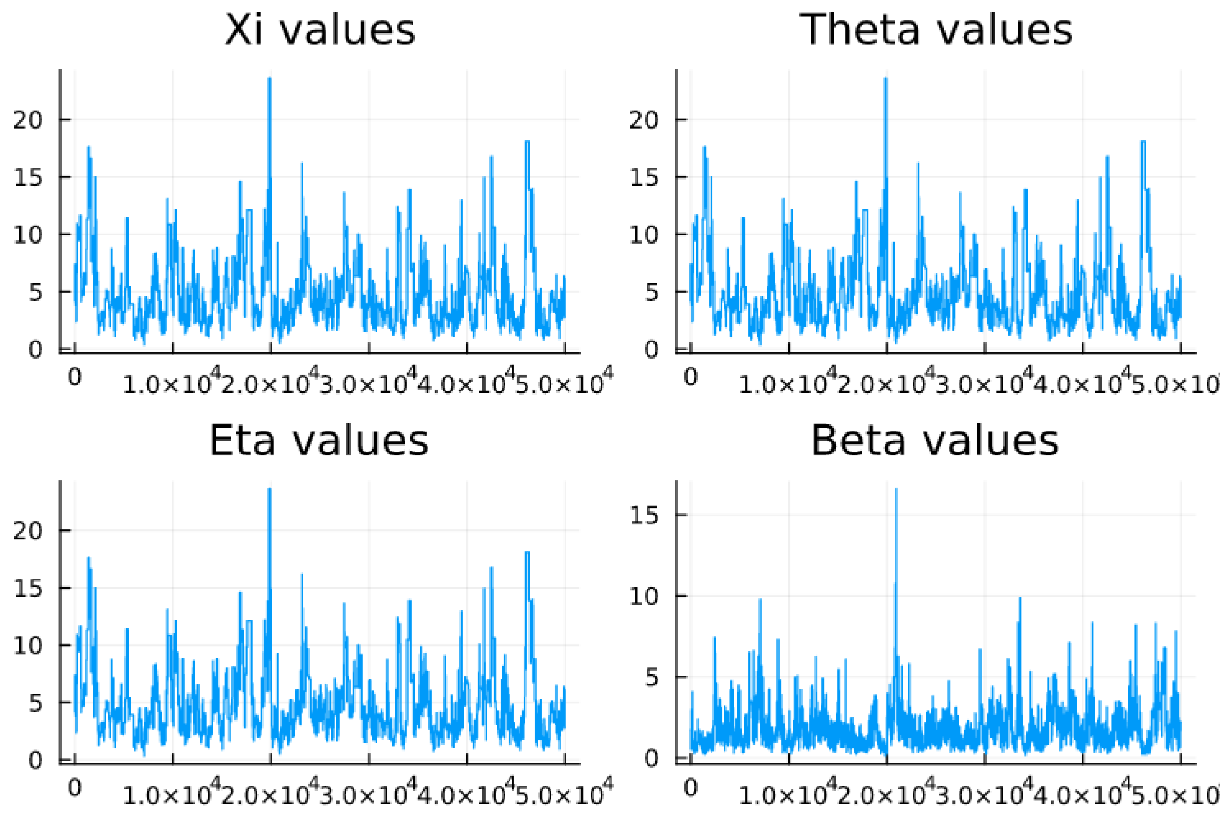
After training the model for 1000 iterations, these are the mean values for the latent variables:

```
Theta mean:2.3426399932849895
Beta mean: 3.106204635158711
```



I re-trained the model using the last trace as an initial trace for 50000 more iterations and obtained the following results:

```
Theta mean:5.03498462034903
Beta mean: 1.8034998410961798
```



References

[1] MÖBIUS, „Kaggle.com," [Interactiv]. Available: https://www.kaggle.com/datasets/arashnic/book-recommendation-dataset?resource=download.

[2] J. M. H. D. M. B. Prem Gopalan, „Scalable Recommendation with Poisson Factorization”.