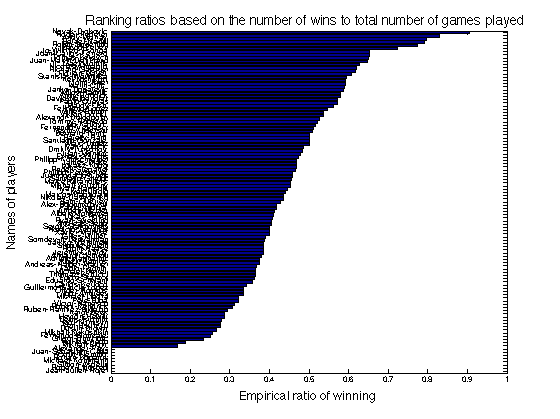
a) Total number of wins and games played of each player in 2011 were computed to calculate their respective ranking ratio. A descending order of this ratios for all the players were plotted based on the bar plot method given. The computation commands written were shown below.

1. **m=zeros(107,2); % The matrix to store total number of wins and played**
2. **m(P,1) = m(P,1) + 1; %Increment number of wins when iterating each row**
3. **m(P,2) = m(P,2) + 1; %Increment number of play when iterating each row**



However, this is not a good way to estimate player skills since this ranking method ignores who each player plays against since this wouldn’t be fair to people that [beat up on far worse players](http://www.codinghorror.com/blog/archives/000961.html) or players who got decimated but maybe learned a thing or two. Defecting a highly skillful player like Andy Murray should earn more credits than defecting a less skillful player. Although, it might make sense but this wouldn’t be fair to people that played a lot (or a little), especially the players at the bottom who only played few games. In addition, it might be because they were usually eliminated by strong players like Roger Federer and did not have chances to play less skillful player to earn some credits.

b) Lines required to sample from the conditional distributions needed for Gibbs iternations were added and they were illusrated below.

**1. m(p) = t’\*((p==G(:,1)) - (p==G(:,2)));% the mean of the conditional skill distribution given the t\_g**

**2. for i = 1:M % the sum of precision matrices contributed by all the games (likelihood terms)**

**for j = 1:i**

**if i==j**

**iS(i,j) = sum(i==G(:,1)) + sum(i==G(:,2)); %Sum all id delta functions if i=j**

**else**

**iS(i,j) = -sum((i==G(:,1)).\*(j==G(:,2))+(i==G(:,2)).\*(j==G(:,1)));**

**iS(j,i) = iS(i,j); %Sum all the negative multiplication of two id delta functions otherwise**

**end**

**end**

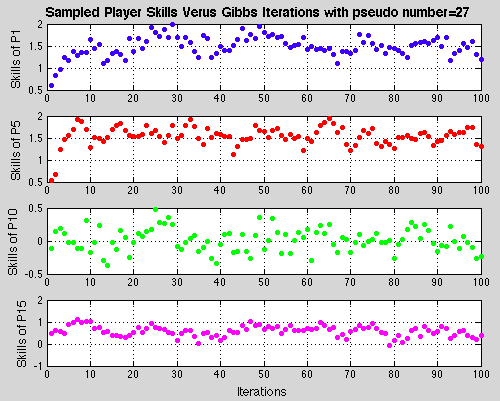
**end**

**%The two central commands were translated from the mean and covariance equation given in page 20.**

c&d) Sampled player skills of Player 1, Player 5, Player 10, Player 15 verus Gibbs sampling running 100 iternations were demonstrated correspondingly in the figure below by blue, red, green and magenta.

1. V(o)=w(1); X(o)=w(5); Y(o)=w(10); Z(o)=w(15);%Store players’ skill. o is the number of iterations.

As can be seen, the Gibbs sampler is able to move around the posterior distribution since the scattered dots of each player for 100 iterations can fluctuate around their mean. Moreover, samples are preferable to be spaced apart by 10 units to be roughly independent otherwise successive samples will be highly correlated and they can slow down Gibbs sampling.



In overall, the Gibbs sampling seems to converge based on the graph shown above because the Gibbs sampling for the four different skills of players roughly reach a stationary distribution in the end.

|  |  |
| --- | --- |
|  |  |

Furthermore, the same player skills as a function of the Gibbs iterations were plotted by using different pseudo random number seeds, which were plotted on the graphs shown above. The left one demonstrated the player skills with pseudo number=15 and the right one illustrated the player skills with pseudo number=50. As can be seen, they roughly reach the same distribution compared with the previous one when pseudo number=27 finally.

e) 100 roughly independent samples from the Gibbs sampler were generated by saving samples spaced by 10 units derived from part c. The central commands for this generation were illustrated on the right below.

|  |  |
| --- | --- |
|  | **% Generating 100 independent samples**  **for o = 1:991**  **if mod(o+9,10)==0 % test if o+9 can be divided by 10**  **r(:,(o+9)/10)=w; %Subsampling by every 10th unit**  **end**  **end**  **% Computing a ranking of players**  **k = mean(r,2); %Calculate the mean of each player’s skill**  **pp = zeros(M,1);**  **for P1=1:M**  **for P2=1:M**  **pp(P1)=pp(P1)+normcdf(k(P1)-k(P2)); %Calculate average probaility**  **end**  **end**  **pp=pp/107;**  **%Sorting and plotting bar chart** |

The central commands for computing a ranking of player by computing the average probability that each player will win against any other player were shown on the right above. As compared with the ranking obtained in part a, the ranking ratios of some players are zero while in the new ranking system all of the players have tangible values. In addition, the probabilities of winning of some highly skillful players like Novak-Djokovic are higher than the ones obtained in part a. First of all, this is because the new ranking system considers the factor of each player who a particular player plays against. For example, defecting a highly skillful player like Andy Murray will earn more credits than defecting a less skillful player. Secondly, the new ranking system computed the order of winning probability based on the skills posterior distribution of players while the ranking in part a computed the order according to the actual empirical results. Finally, the generation of 100 independent spaced samples also enhances the speed of Gibbs sampling for ranking in part e.

f) Based on the samples generated in part e, the probabilities that each of the 4 top players wins against each other were illustrated in the table below. The second row represents the player who wins the game while the first column stands for the player who loses the corresponding game. However, the probability of each player playing against himself is ignored since this situation cannot happen in reality.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Probabilities that each player wins against each other (Loser\Winner)** | | | | |
|  | Novak-Djokovic | Roger-Federer | Rafael-Nadal | Andy-Murray |
| Novak-Djokovic | - | 0.3754 | 0.3492 | 0.3063 |
| Roger-Federer | 0.6246 | - | 0.4721 | 0.4251 |
| Rafael-Nadal | 0.6508 | 0.5279 | - | 0.4527 |
| Andy-Murray | 0.6937 | 0.5749 | 0.5473 | - |

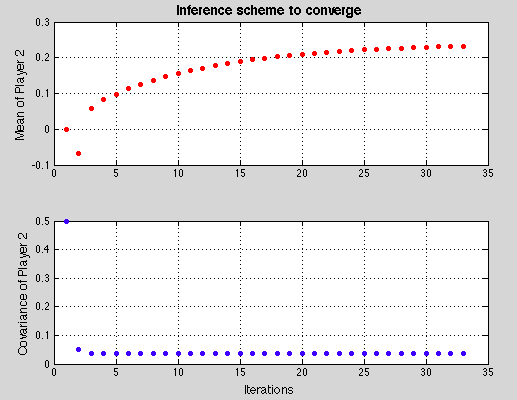
The central commands for sorting and computation of probability were demonstrated below.

|  |  |
| --- | --- |
| **k = mean(r,2);**  **[kk,ii] = sort(k, 'descend');**  **for P=1:4**  **skills(P)=kk(P); %The skills of each player**  **player(P)=W(ii(P)); %The name of each player**  **end** | **for P1=1:4**  **for P2=1:4**  **if P1 ~= P2 %Only return if they are different**  **[player(P1) 'VS' player(P2)]%Names**  **normcdf(skills(P1)-skills(P2))%Winning P**  **end**  **end**  **end** |

g) Do inference for the True Skills model by running message passing and EP. The essential number of iterations for inference model to converge was approximated by applying the central commands below.

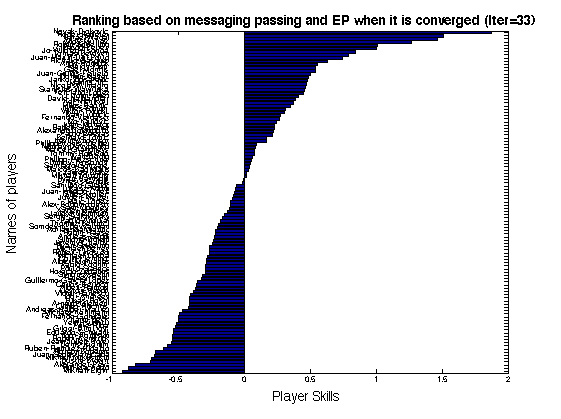
|  |  |
| --- | --- |
| **V1(:,iter)=Ms; %Mean matrix per iternation**  **V2(:,iter)=1./Ps; %Covariance matrix per iternation**  **if iter > 5**  **if (abs(V1(1,iter)-V1(1,iter-1))<0.001) && (abs(V2(1,iter)-V2(1,iter-1))<0.001)%Test if converge**  **NN = iter;%Record iteration that start converging**  **break; %Break the loop**  **end**  **end** | **subplot(2,1,1);**  **plot(V1(2,:),'r.');**  **subplot(2,1,2);**  **plot(V2(2,:),'b.');** |

By computing the absolute difference value between the current mean and covariance and previous mean and covariance, the convergence occurs when these two absolute values are smaller than 0.001. As a result, the number of iterations for this inference model that is going to start converging is 33. As an example, the mean (red) and covariance (blue) of player 2 to converge after 33 iterations were shown below.



h) The central commands for producing a ranking after 33 iterations for convergence were shown below.

|  |  |
| --- | --- |
| **pp = zeros(M,1);**  **pp = V1(:,NN);**  **%Ranking based on the results of part g**  **[kk, ii] = sort(pp, 'descend');** | **np = 107;**  **barh(kk(np:-1:1)) %Plot bar chart for ranking**  **set(gca,'YTickLabel',W(ii(np:-1:1)),'YTick',1:np,'FontSize',6)**  **axis([-1 2 0.5 np+0.5]) %Min and max x-axis are -1 and 2** |



Compared with the ranking in part e and part a, the main difference is that the new ranking system has both positive and negative values for 107 players. In addition, it only has the approximate zero value in the middle range of all the players and it also has a much higher value than the previous two graph at the two extreme of the ranking range. This is because the new ranking computes the actual converged mean skill value rather than the winning probability against other player, which is only ranged from 0 to 1 for the previous two graphs. The ranking method using messaging passing and EP is best since it gives us a scale where each person has skill value expressing their rating that we can use for comparison and it is also direct way for us to see who’s better than whom and by how much. If a player has a skill rating much higher than someone else, we would expect them to win if they played each other. Moreover, we can also use this method to predict the probability of winning and losing more accurately than other two ways. Furthermore, Gibbs sampling usually has correlation issues to be considered for fast sampling and it is also very time-consuming to compute ranking if we have a huge amount of data.

i) The probabilities that each of the 4 top players wins against each other were computed based on message passing and EP. The central commands for sorting and computation of probability were demonstrated below.

|  |  |
| --- | --- |
| **pp = zeros(M,1);**  **pp = V1(:,NN);**  **[kk, ii] = sort(pp, 'descend');**  **for P=1:4**  **skills(P)=kk(P); %The skills of each player**  **player(P)=W(ii(P)); %The name of each player**  **end** | **for P1=1:4**  **for P2=1:4**  **if P1 ~= P2 %Only return if they are different**  **[player(P1) 'VS' player(P2)]%Names**  **normcdf(skills(P1)-skills(P2))%Winning P**  **end**  **end**  **end** |

The probabilities of winning against each other were illustrated below. The second row represents the player who wins the game while the first column stands for the player who loses the respective game. However, the probability of each player playing against himself is ignored since this situation cannot happen in reality.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Probabilities that each player wins against each other (Loser\Winner)** | | | | |
|  | Novak-Djokovic | Roger-Federer | Rafael-Nadal | Andy-Murray |
| Novak-Djokovic | - | 0.3571 | 0.3395 | 0.2729 |
| Roger-Federer | 0.6429 | - | 0.4810 | 0.4060 |
| Rafael-Nadal | 0.6605 | 0.5190 | - | 0.4245 |
| Andy-Murray | 0.7271 | 0.5940 | 0.5755 | - |

Compared with the results obtained in part f, the winning probabilities of more highly skillful players such as Novak-Djokovic and Roger-Federer against less skillful players such as Rafael-Nadal and Andy-Murray according to the ranking graph in part h are a little higher than the results gained in part f. It is concluded that the ranking system using message passing and EP is more certain on the prediction of winning or losing by computing marginal skills distribution of each player because the factor nodes on the message passing network can help simplify further marginal conditional skills accurately and optimize efficiency, which can also be justified by the convergence graph in part g.

j) The central commends added in the part I were shown below.

1. [player(P1) 'VS' player(P2)]

2. normcdf(skills(P1)-skills(P2))/normcdf(skills(P1)) %Posterior distribution/Prior = Probability given skills.

The probabilities that the player has a higher skill for part f and part i were illustrated in the following table.

The probability that the player has a higher skill is computed by the posterior obtained in the previous part divided by the prior skill.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Part F - Probabilities that each player has a higher skill (Lower\Higher)** | | | | |
|  | Novak-Djokovic | Roger-Federer | Rafael-Nadal | Andy-Murray |
| Novak-Djokovic | - | 0.4025 | 0.3783 | 0.3386 |
| Roger-Federer | 0.6471 | - | 0.5114 | 0.4700 |
| Rafael-Nadal | 0.6743 | 0.5659 | - | 0.5005 |
| Andy-Murray | 0.7187 | 0.6163 | 0.5929 | - |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Part I - Probabilities that each player has a higher skill (Lower\Higher)** | | | | |
|  | Novak-Djokovic | Roger-Federer | Rafael-Nadal | Andy-Murray |
| Novak-Djokovic | - | 0.3821 | 0.3657 | 0.3037 |
| Roger-Federer | 0.6630 | - | 0.5182 | 0.4519 |
| Rafael-Nadal | 0.6811 | 0.5553 | - | 0.4725 |
| Andy-Murray | 0.7498 | 0.6356 | 0.6200 | - |

Compared with the data in these two tables, they demonstrate the similar patterns as we obtained in the previous part for computing the probability that each player can win against each other. The probabilities computed by message passing and EP method generally show higher values than the ones obtained by Gibbs sampling, which further justify message passing and EP is more certain to predict the probability of prior, likelihood and posterior.